KDD 2017
Halifax, Nova Scotia - Canada
August 13 - 17, 2017

Tutorial: Data-Driven Approaches towards Malicious Behavior Modeling

Meng Jiang
University of Notre Dame

Srijan Kumar
Stanford University

Christos Faloutsos
Carnegie Mellon University

V.S. Subrahmanian
University of Maryland, College Park

Tutorial link: http://bit.ly/kdd2017

Outline

Introduction

Feature-based algorithms

- Sockpuppets
- Vandals
- Hoaxes

Spectral-based algorithms

- Visualization: “spokes”, “blocks”, “staircases”
- Camouflage
- Theoretical guarantee

Density-based algorithms

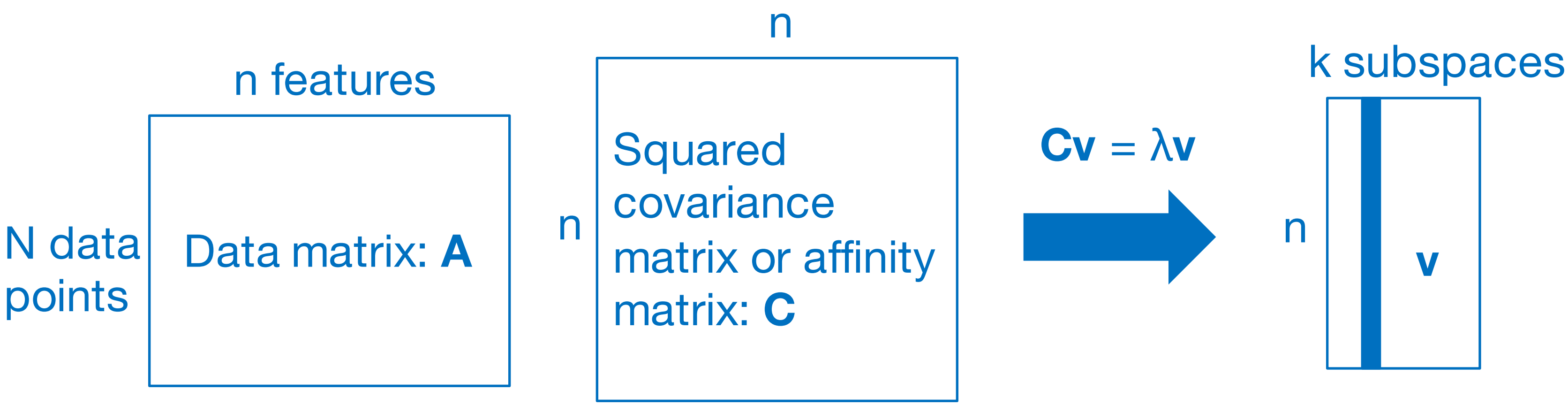
- Ill-gotten Likes
- Synchronized Behaviors
- Advertising campaigns
- Social spam

Conclusions and future directions

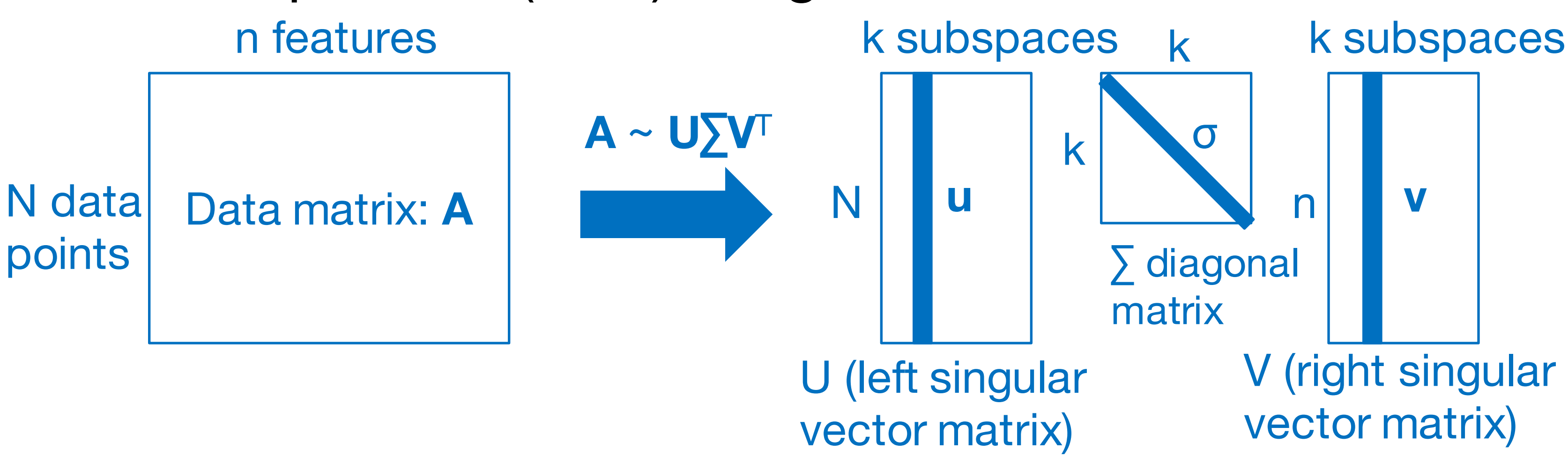
Tutorial link: http://bit.ly/kdd2017

How to get spectral subspaces?

- Frequency components → Principal Component Analysis (PCA): Eigenvectors



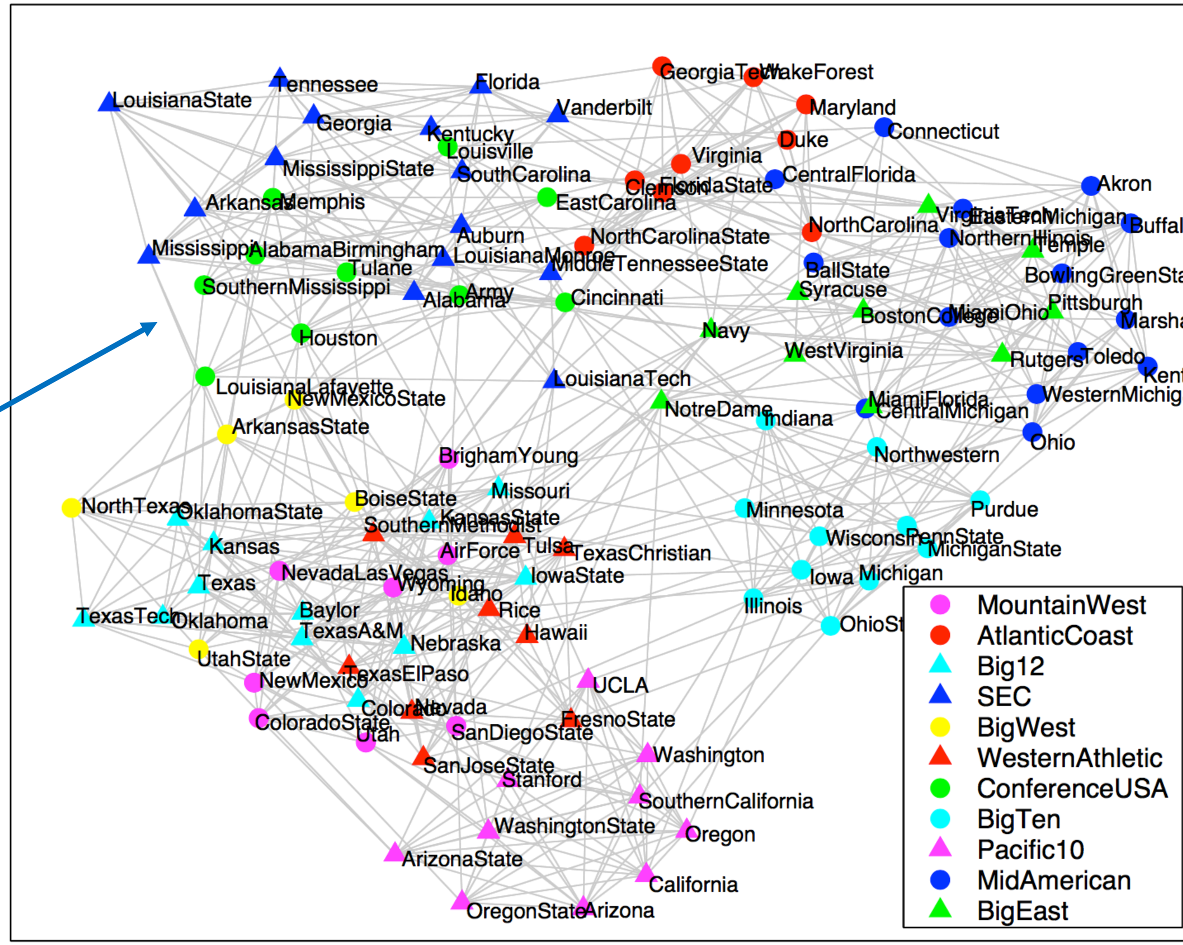
- Other spectral decomposition methods. Singular Value Decomposition (SVD): Singular vectors



Spectral methods: Community Identification

Nodes are USA college football teams and edges represent which team played with which other team.

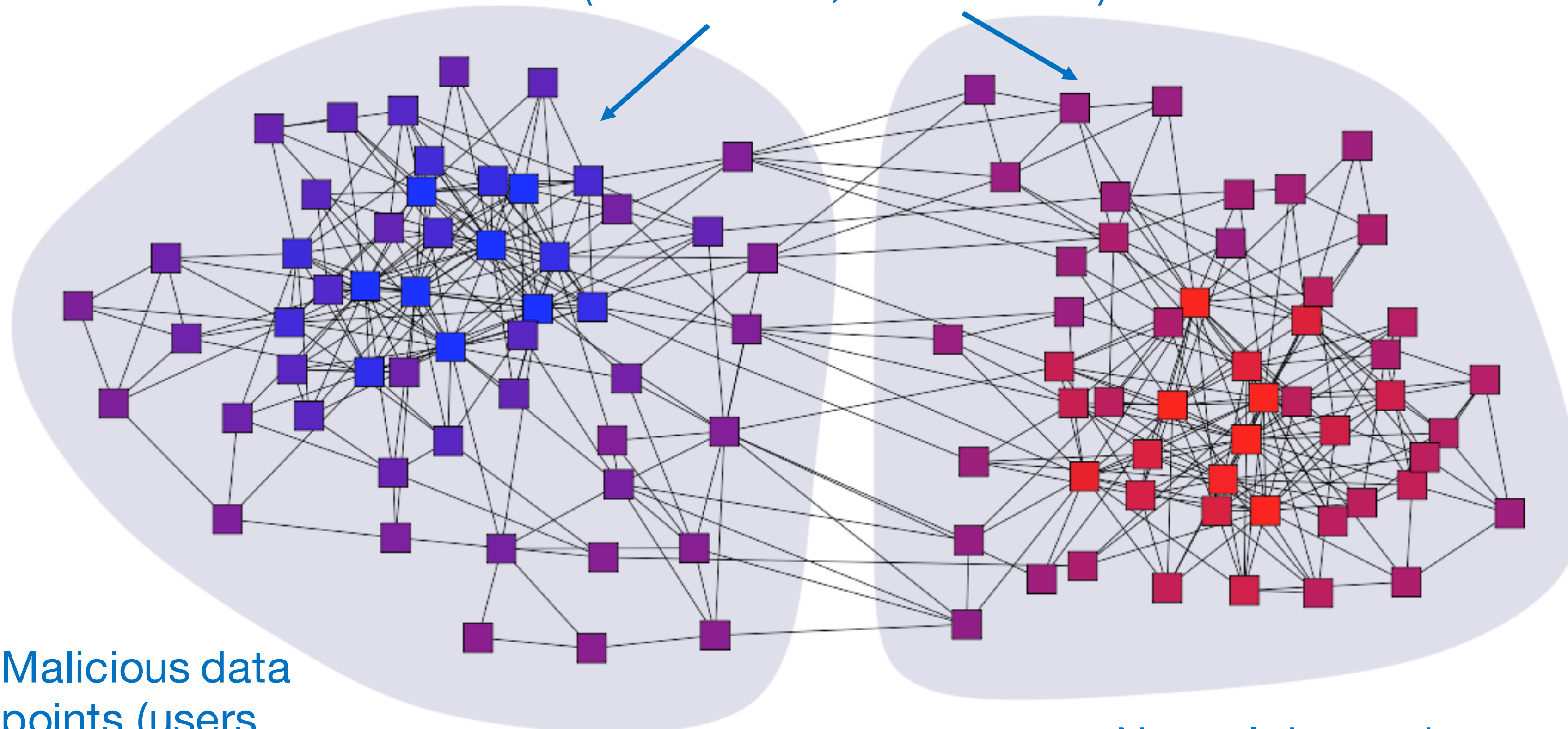
Communities represent groups of frequently co-playing teams.



Scott White and Padhraic Smyth. “A spectral clustering approach to finding communities in graphs”, SDM 2005.

Spectral methods: Anomaly detection

Leading eigenvector
(blue for one, red for other)



Malicious data points (users, behaviors, communities, etc.)

Normal data points

Potential application: anomaly detection

M. E. J. Newman. "Finding community structure in networks using the eigenvectors of matrices", Physical Review E 2006.

Spectral-based methods

- Advantages
  - Visualization: tunable value of k = number of subspaces
  - Feature extraction by data distribution rather than manual or automatic selection
  - “Principal” components represent “leading” vectors
  - Data: Can easily work with N-by-N graphs, N-by-N-by-N tensors
  - Applications: Finding communities and anomalies
- Disadvantages
  - Lack of interpretability of the subspaces/features

Finding Surprising Spectral Patterns in Large Graphs

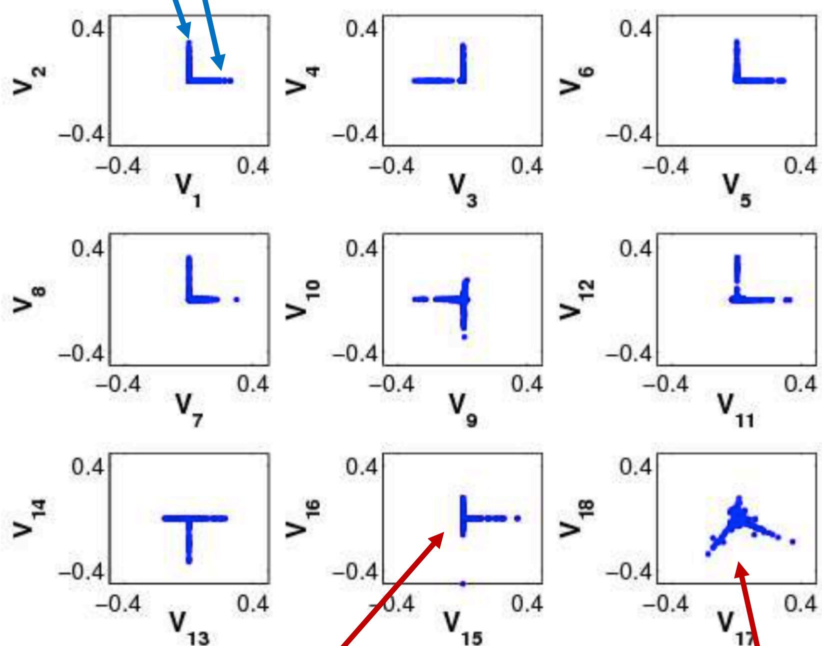
- Problem definition: Given a social graph based on mobile calls made from/to callers, find caller communities.
- Dataset: Activity over the duration of a month, 186,000 nodes and 464,000 edges.
- Key contribution: Discovery of the "spokes" phenomenon
  - **The singular vectors of the graph, when plotted against each other, often have clear separate lines, typically aligned with axes.**
  - Use EigenEigen (EE) plots to identify communities in the form of **cliques or near-cliques**, **perfect or near-perfect bipartite-cores**.

B. Aditya Prakash, Mukund Seshadri, Ashwin Sridharan, Sridhar Machiraju, Christos Faloutsos. "EigenSpokes: Surprising Patterns and Scalable Community Chipping in Large Graphs", PAKDD 2010.

Spokes and Dense Cliques

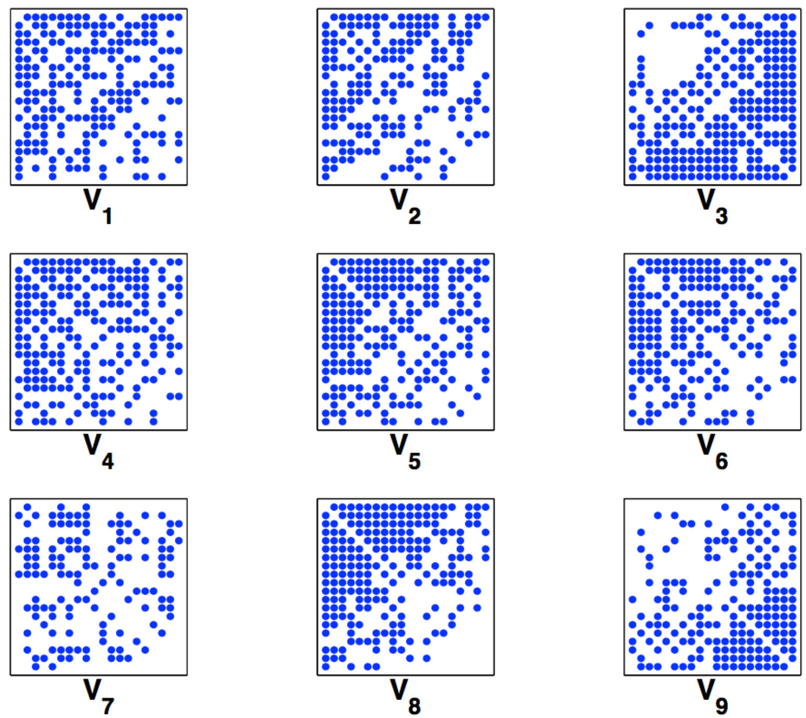
V-V plots: Right spectral subspaces

Spokes



(a) *EE-plot*

Subgraphs of nodes identified as important by different V vectors: all are tightly knit



(b) Spy Plots of Sub-graph of Top 20 Nodes

Long vs. Short?

Tilted?

Prakash, et al. (PAKDD 2010)

Spectral subspace

- What is the meaning of spokes, elongated spokes, tilted spokes?
- Are there other patterns?
- Can these patterns be used to identify malicious behavior?

Inferring Lockstep Behavior from Connectivity Patterns

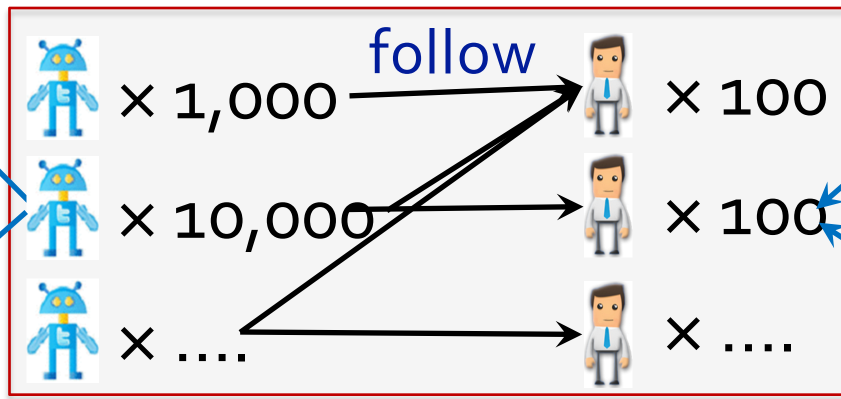
- Problem definition: Given a large graph, from **spectral subspace plots**, can we infer **lockstep behavior** patterns?

Meng Jiang, Peng Cui, Alex Beutel, Christos Faloutsos, Shiqiang Yang. "Inferring lockstep behavior from connectivity pattern in large graphs", KAIS 2016.

Inferring Lockstep Behavior from Connectivity Patterns

- Problem definition: Given a large graph, from **spectral subspace plots**, can we infer **lockstep behavior** patterns?

“Camouflage”: fraudsters follow other users to hide their behavior

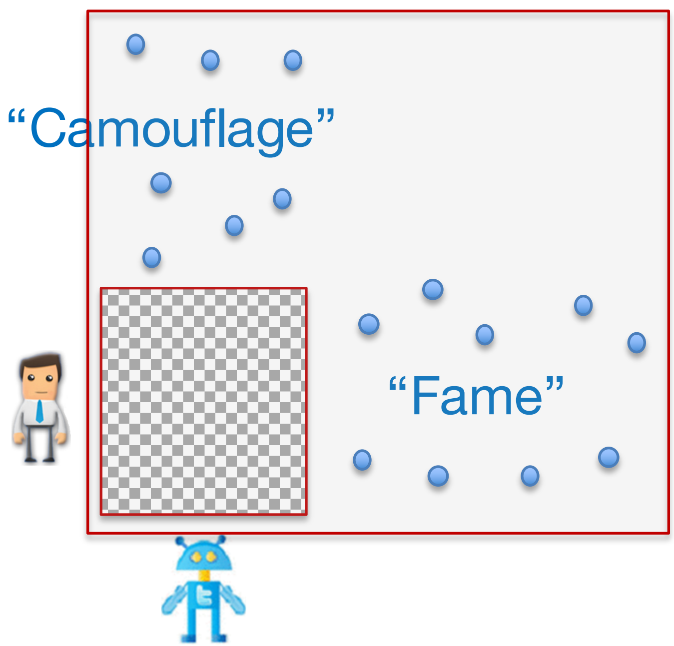


“Fame”: popular users are followed by several users

Jiang, et al. (KAIS 2016)

Inferring Lockstep Behavior from Connectivity Patterns

- Problem definition: Given a large graph, from **spectral subspace plots**, can we infer **lockstep behavior** patterns?

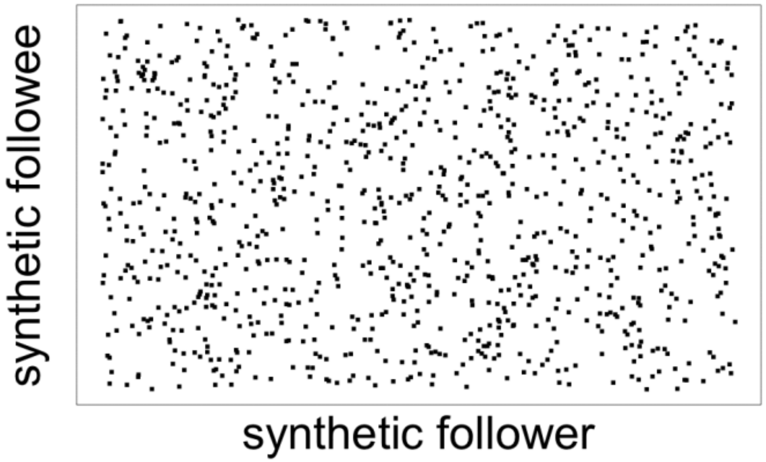


Jiang, et al. (KAIS 2016)

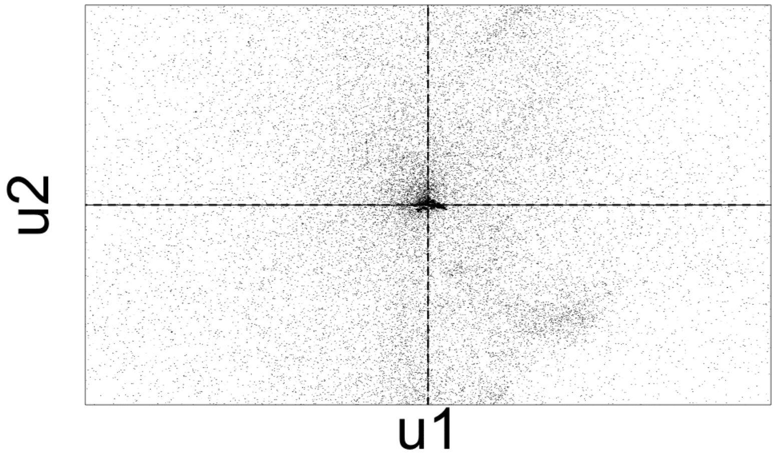
Case 0: No lockstep behavior

- No blocks in adjacency matrix lead to scattering and no patterns in spectral subspace

Adjacency Matrix



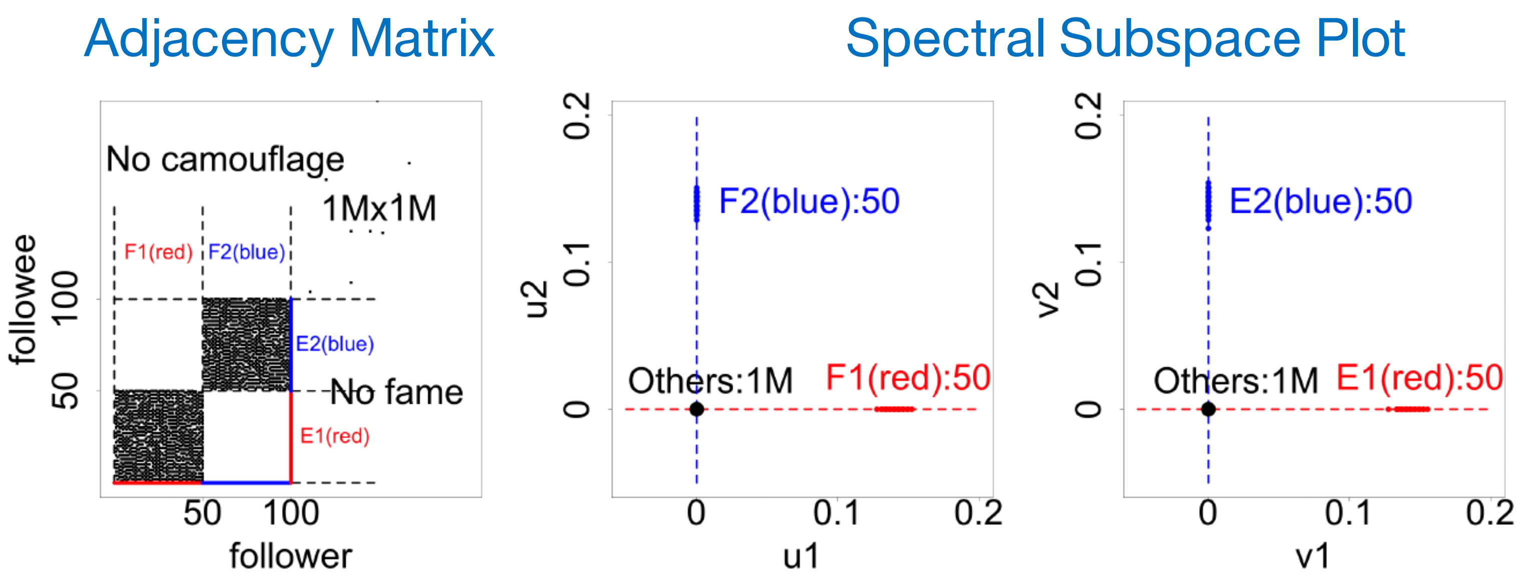
Spectral Subspace Plot



Jiang, et al. (KAIS 2016)

Case 1: Non-overlapping dense lockstep

- Dense blocks in adjacency matrix generates “rays” in spectral subspace

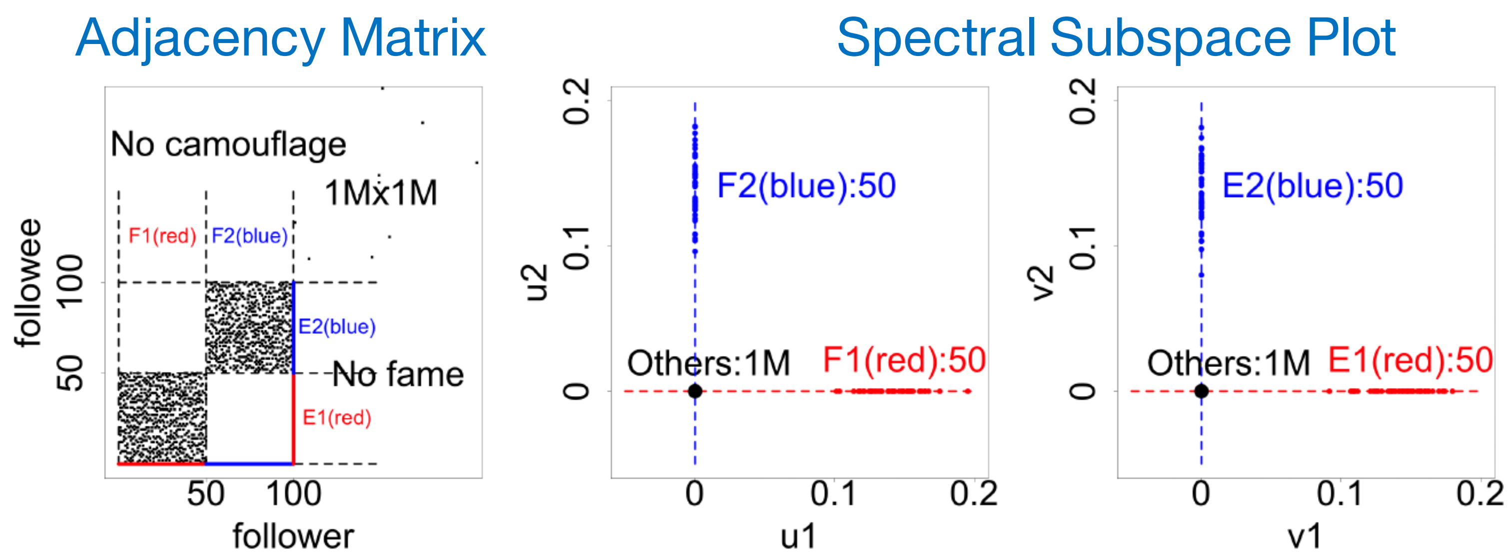


Rule 1 (short “rays”): two blocks, high density $\boxed{(90\%)}$, no “camouflage”, no “fame”

Jiang, et al. (KAIS 2016)

Case 2: Non-overlapping sparse lockstep

- Low density blocks in adjacency matrix leads to elongation of rays, indicating more varied behavior



Rule 2 (long "rays"): two blocks, low density $\boxed{(50\%)}$, no "camouflage", no "fame"

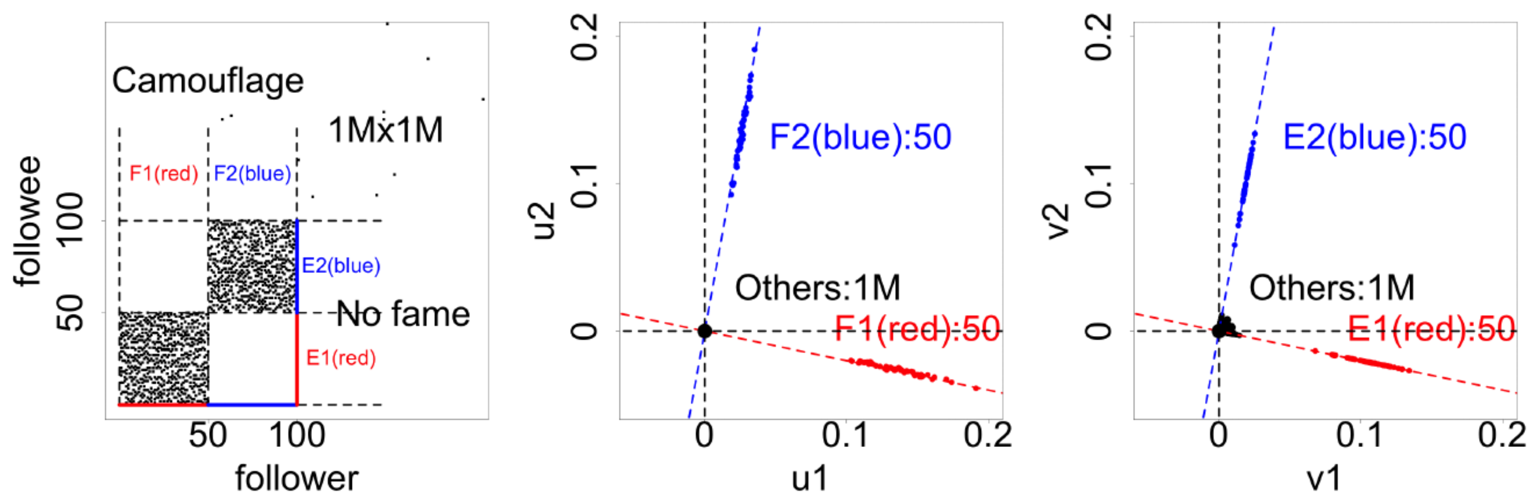
Jiang, et al. (KAIS 2016)

Case 3: Non-overlapping lockstep with outside edges

- Edges to or from blocks in adjacency matrix leads to tilting of rays in spectral subspace
- Edges going out of block: "camouflage" by fraudsters
- Edges into the block: "fame" edges to popular users

Adjacency Matrix

Spectral Subspace Plot



Rule 3 (tilting "rays"): two blocks, with "camouflage", no "fame"

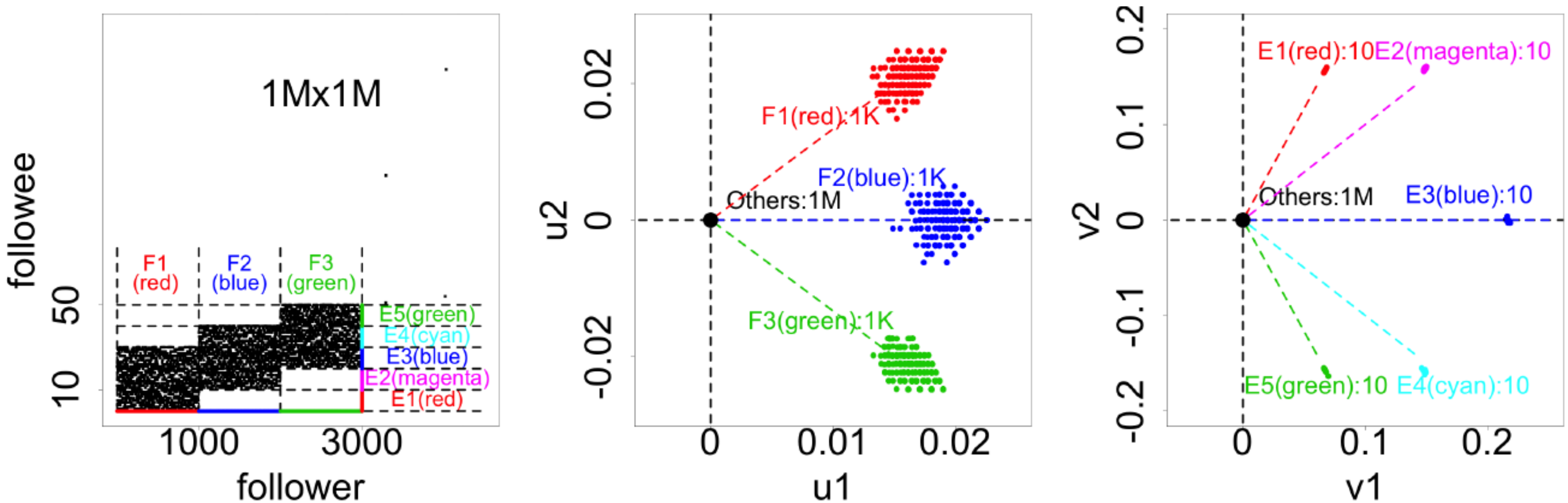
Jiang, et al. (KAIS 2016)

Case 4: Overlapping lockstep

- "Staircase", i.e. sequentially overlapping blocks, in adjacency matrix generates "Pearls" in spectral subspace

Adjacency Matrix with staircase

Spectral subspace plot showing pearls

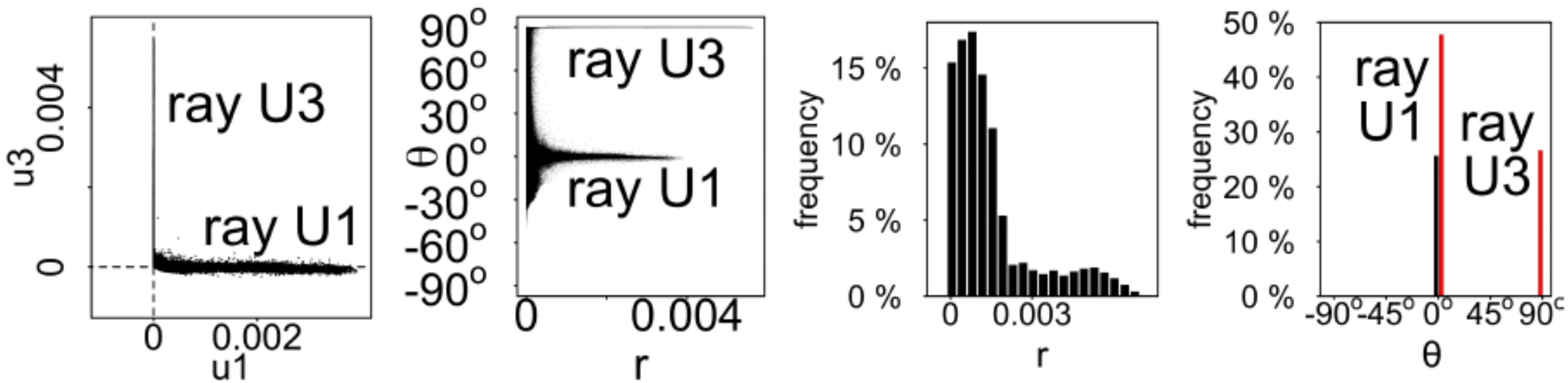


Rule 4 ("pearls"): a "staircase" of three partially overlapping blocks.

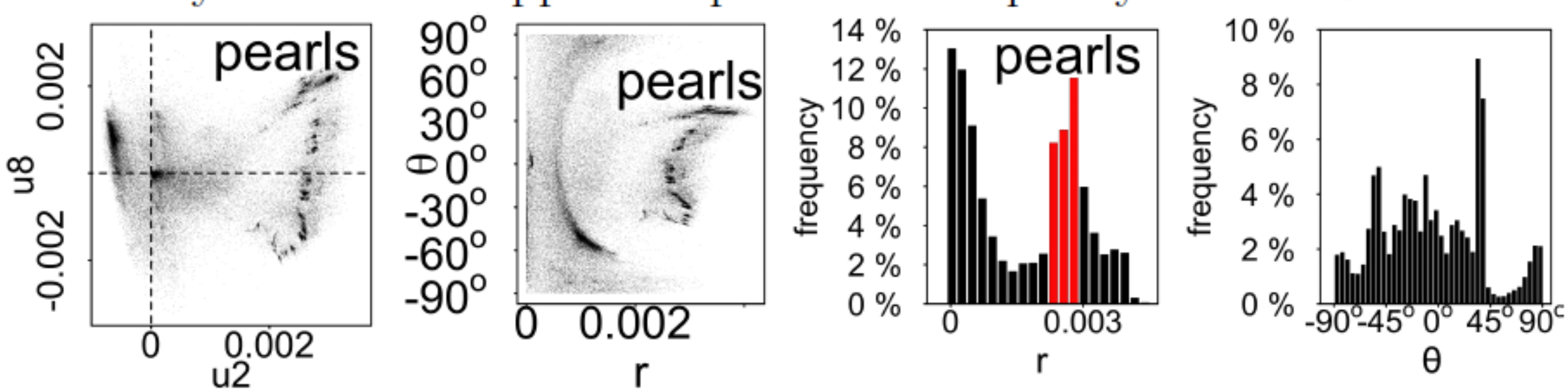
Jiang, et al. (KAIS 2016)

LockInfer Algorithm:
Reading Spectral Subspace Plots

Spectral Subspace Plot | **Polar Coordinate Transform** | **Histograms**



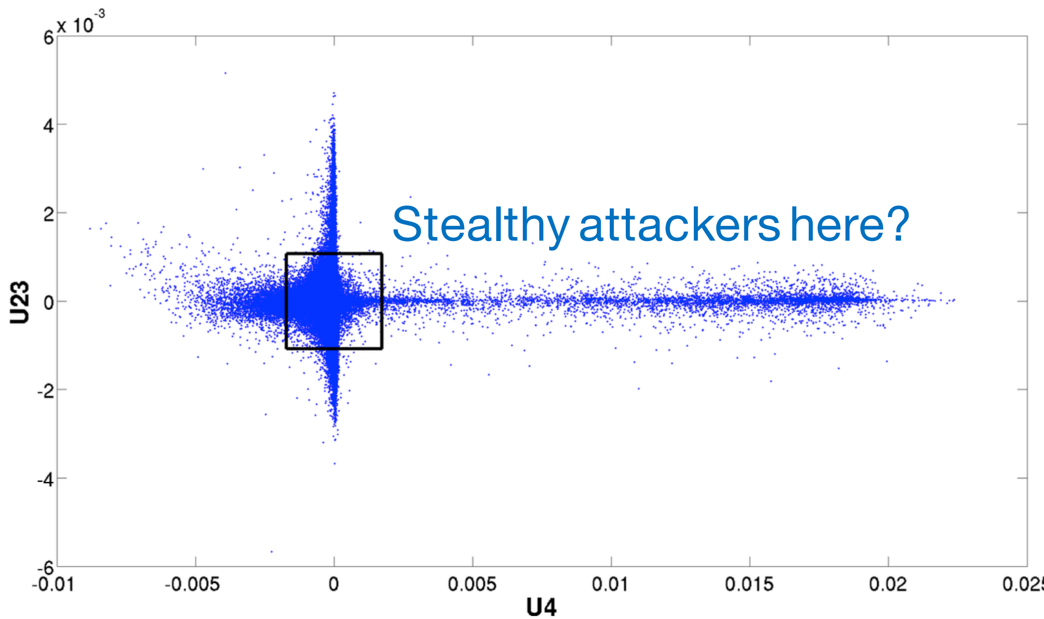
"rays" show two apparent spikes on θ frequency at 0^o and 90^o



"pearls" show a spike on r frequency at a much-greater-than-zero value

Spotting Small-Scale, Stealthy Attacks

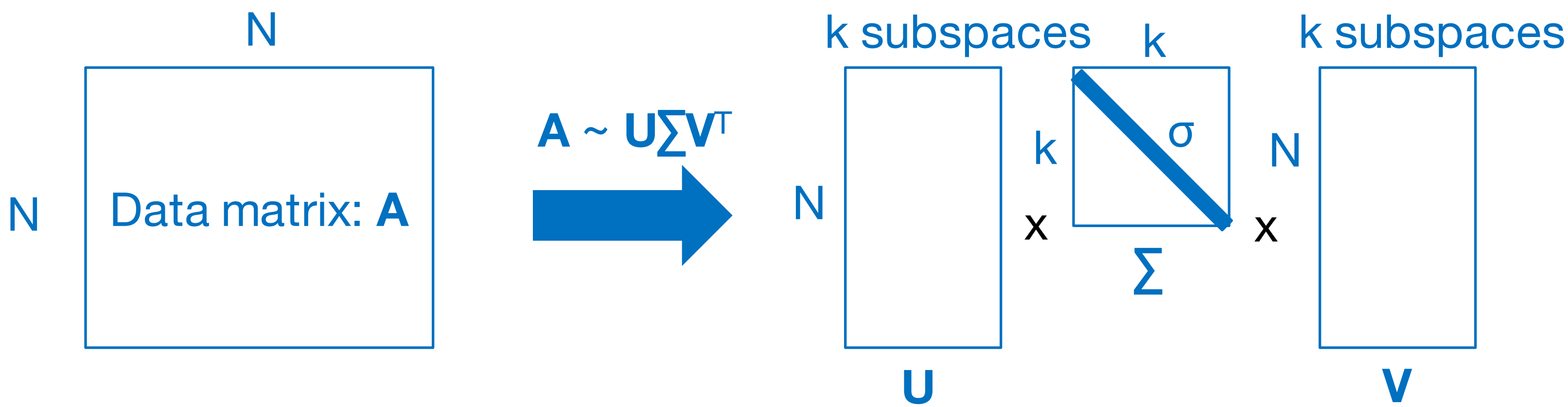
- Problem definition: Can we catch stealthy attacks that are missed by traditional spectral methods?
- Dataset: Twitter "who-follows-whom" social graph, 41.7 million nodes, 1.5 billion edges



Neil Shah, Alex Beutel, Brian Gallagher, Christos Faloutsos. "Spotting suspicious link behavior with fBox: An adversarial perspective", ICDM 2014.

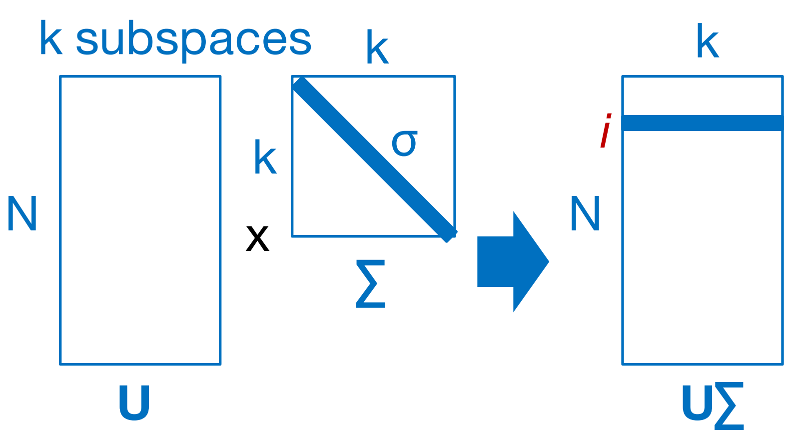
fBox: Reconstructed Degrees

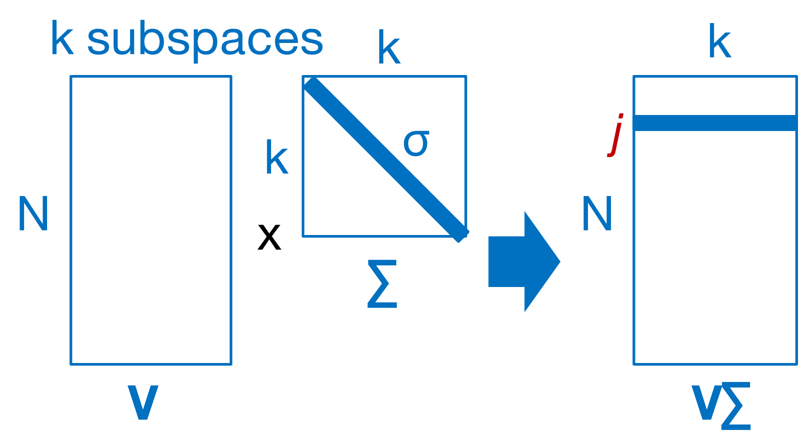
SVD



Reconstructed out-degree(i) = $\|(\mathbf{U}\Sigma)_i\|_2$

Reconstructed in-degree(j) = $\|(\mathbf{V}\Sigma)_j\|_2$





Shah, et al. (ICDM 2014)

Norm-Preserving Property of SVD

- The row vectors of a full rank decomposition and associated projection will retain the same *L2* norm or vector length as in the original space:
  - For k = rank($\mathbf{A}$), $||\mathbf{A_i}||_2 = ||(\mathbf{U}\Sigma)_\mathbf{i}||_2$ and $||\mathbf{A^T}_\mathbf{j}||_2 = ||(\mathbf{V}\Sigma)_\mathbf{j}||_2$

- So, compare:
  - Reconstructed out-degree vs. real out-degree
  - Reconstructed in-degree vs. real in-degree

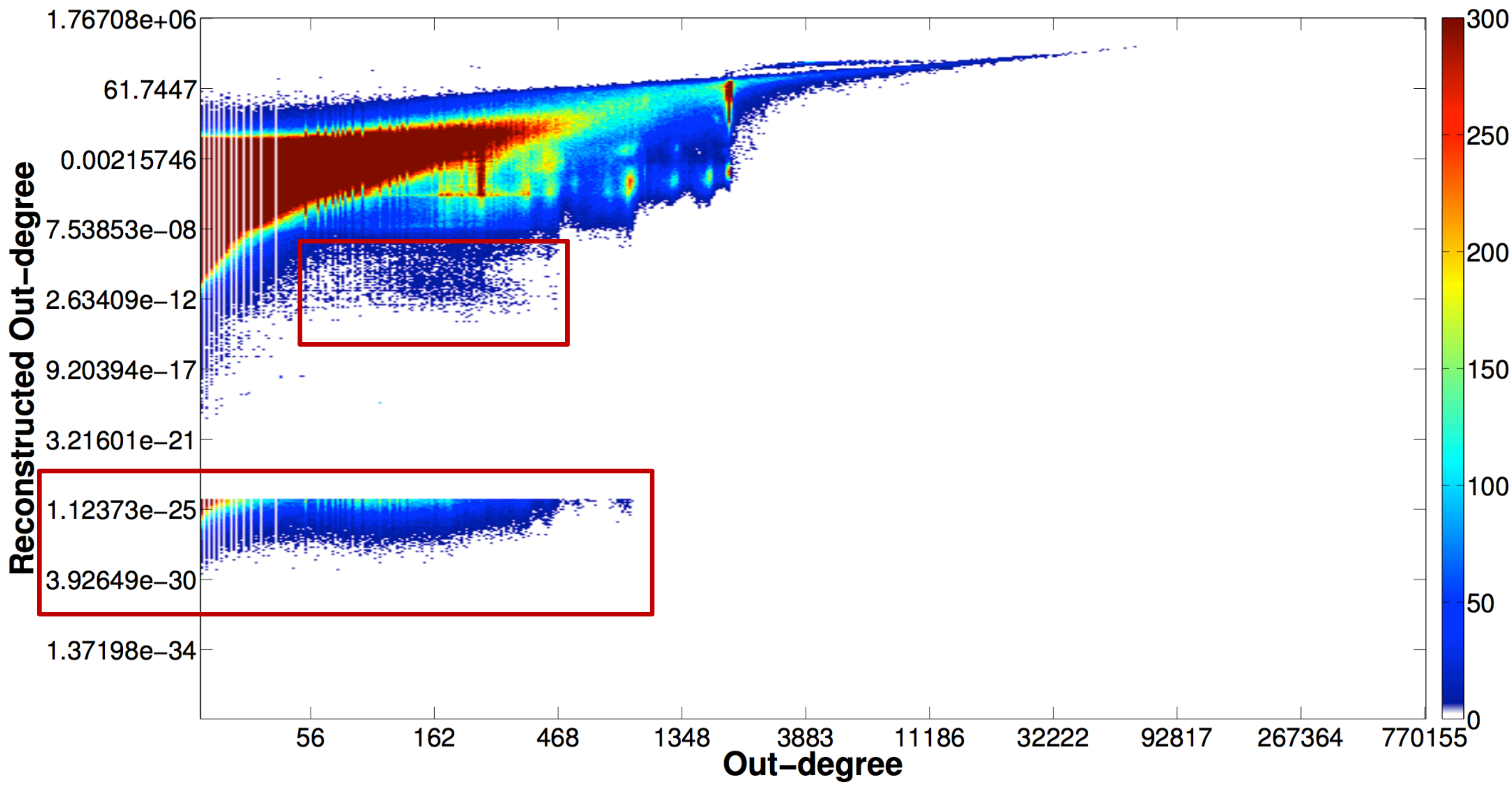
Shah, et al. (ICDM 2014)

Why does fBox work?

For k < rank(A), dishonest users' reconstruction is poor compared to that of honest users.

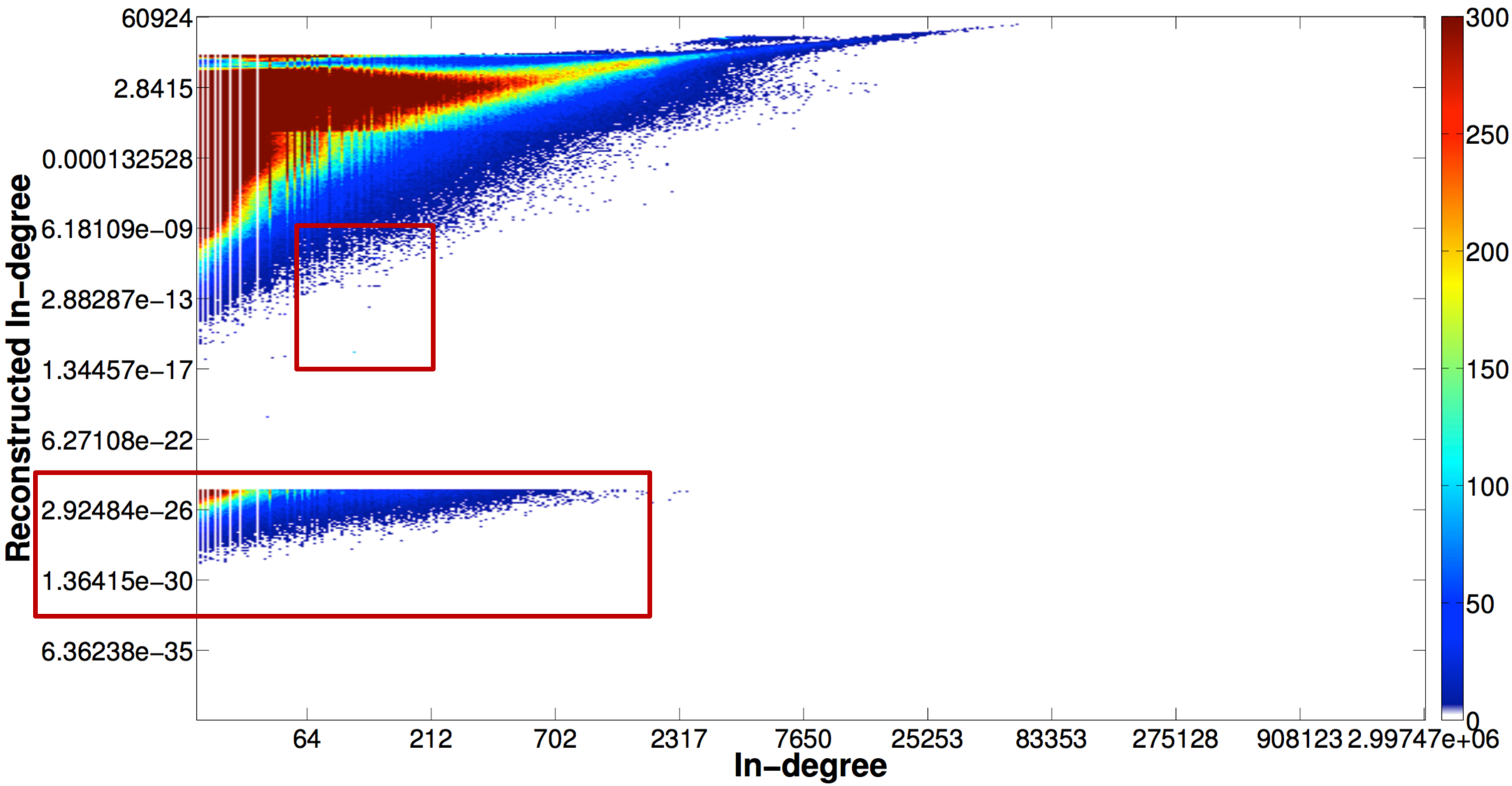
- Dishonest users who either form isolated components or link to dishonest objects will project poorly and have characteristically low reconstruction degrees
- Honest users who are well-connected to real products and brands should project more strongly and have characteristically higher reconstruction degrees

Reconstructed out-degree vs. Real out-degree



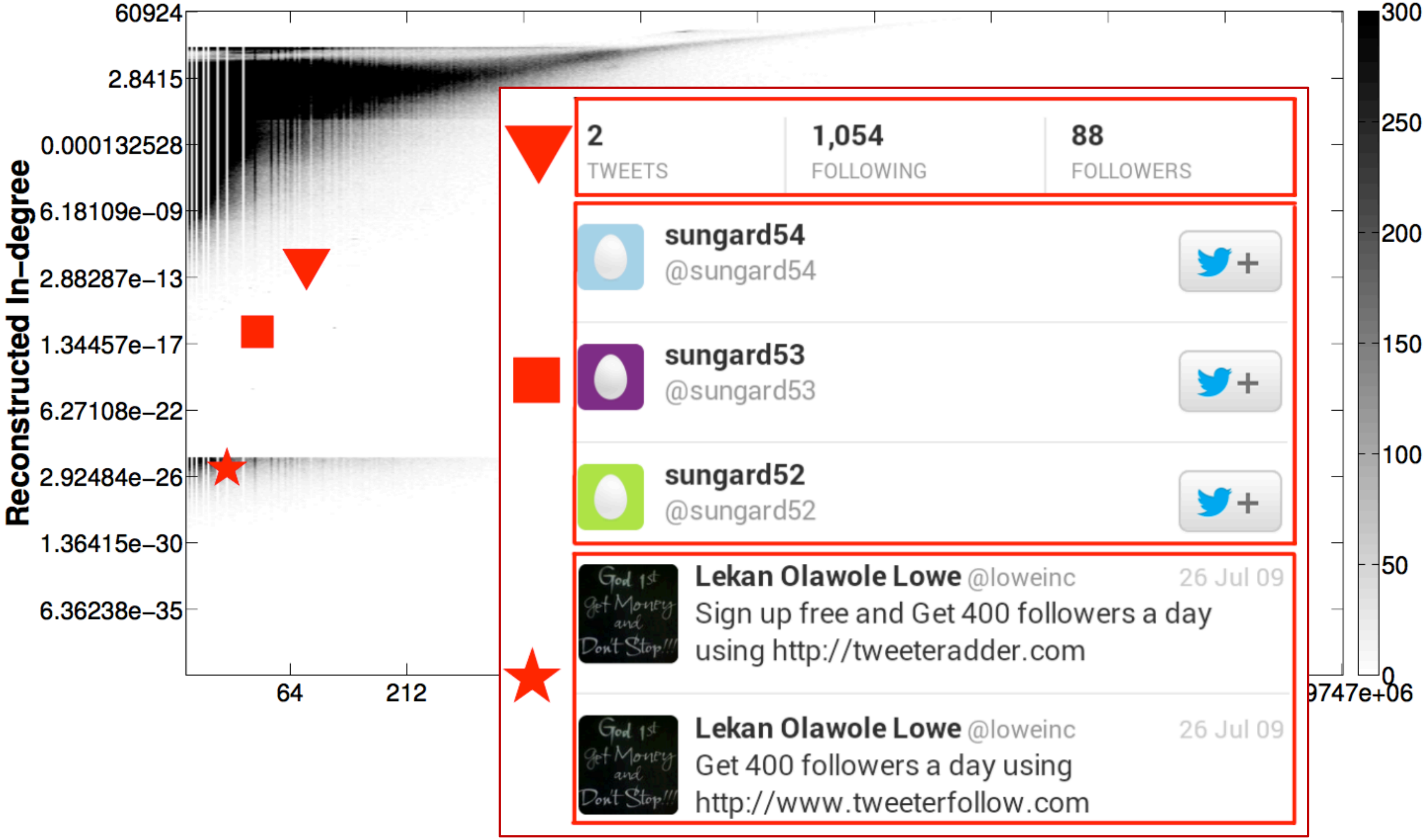
Shah, et al. (ICDM 2014)

Reconstructed in-degree vs. Real in-degree



Shah, et al. (ICDM 2014)

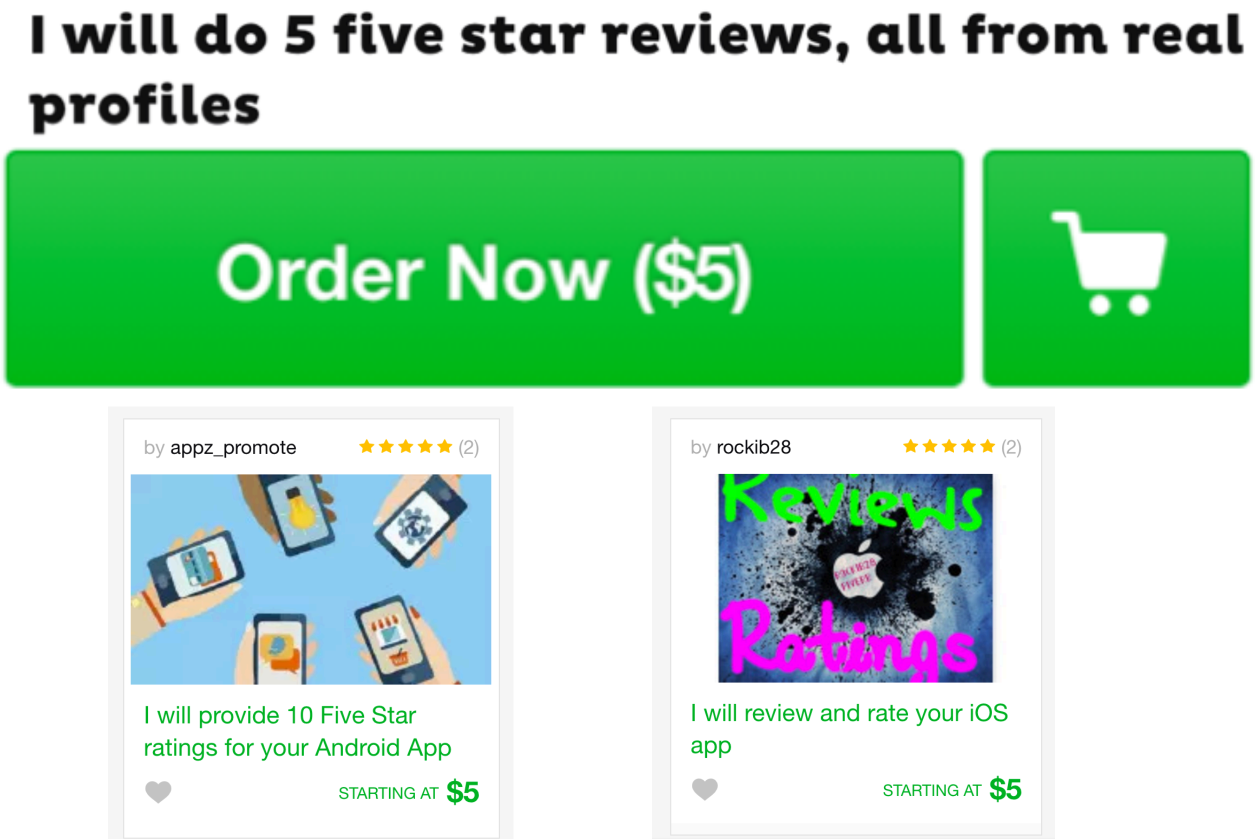
Reconstructed in-degree vs. Real in-degree (cont.)



Shah, et al. (ICDM 2014)

Bounding Graph Fraud in Camouflage

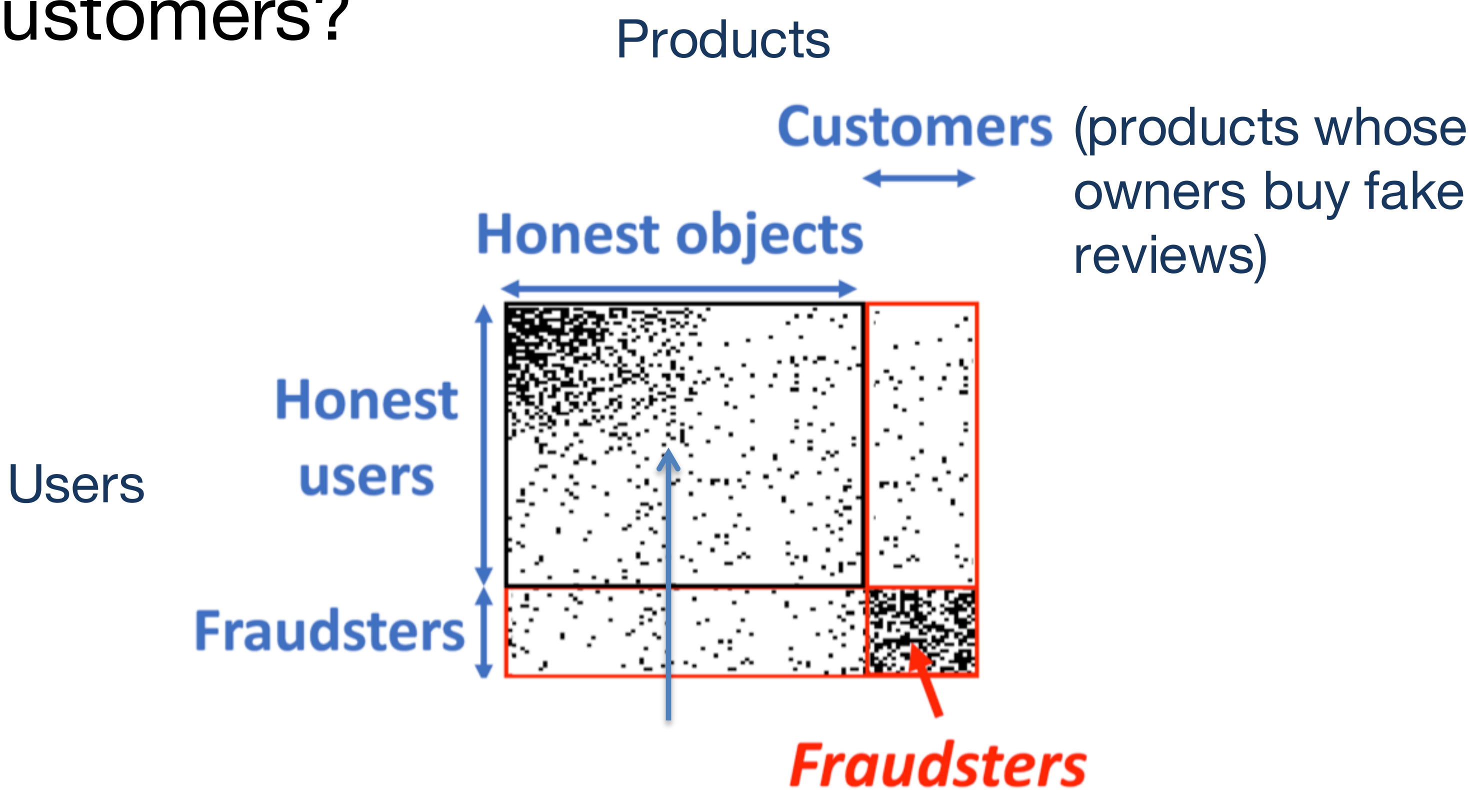
- An application: Fake reviews



Bryan Hooi, Hyun Ah Song, Alex Beutel, Neil Shah, Kijung Shin, Christos Faloutsos. “FRAUDAR: Bounding Graph Fraud in the Face of Camouflage”, **KDD 2016 Best Research Paper Award**.

“User-Product” Review Graph

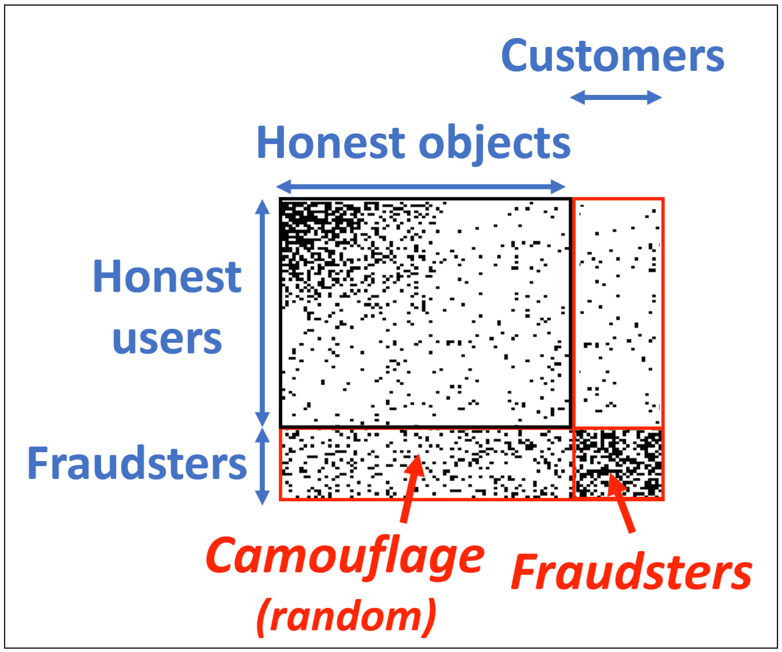
- Problem definition: Given a “user-product” review graph, can we spot fraudsters and customers?



Hooi, et al. (KDD 2016)

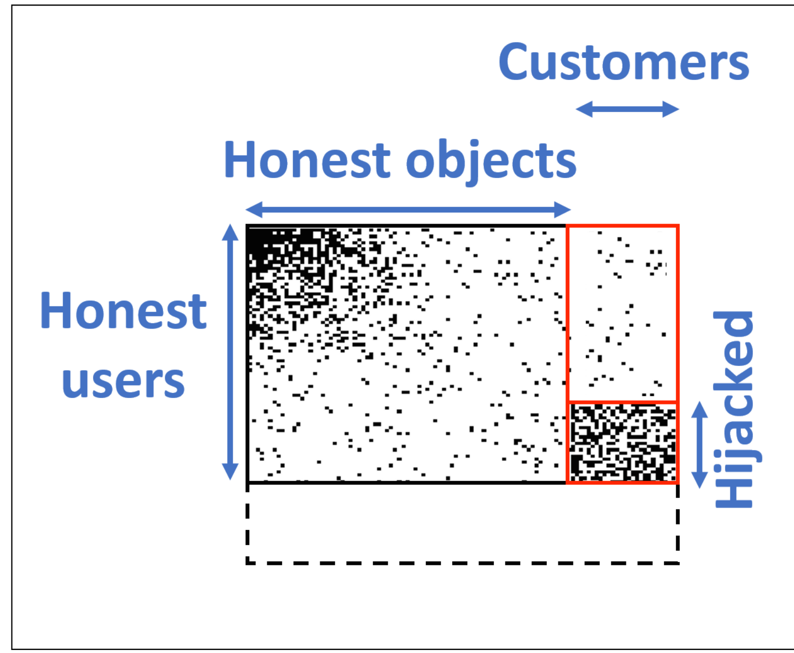
Camouflage: Evading Detection

Random camouflage



"camouflage" in LockInfer

Hijacked user accounts

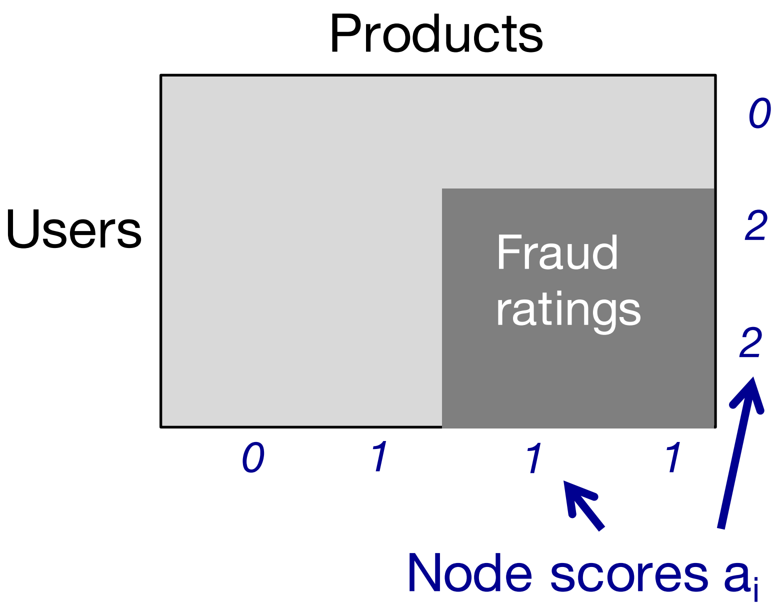


"Fame" in LockInfer

Hooi, et al. (KDD 2016)

Formal Problem Definition

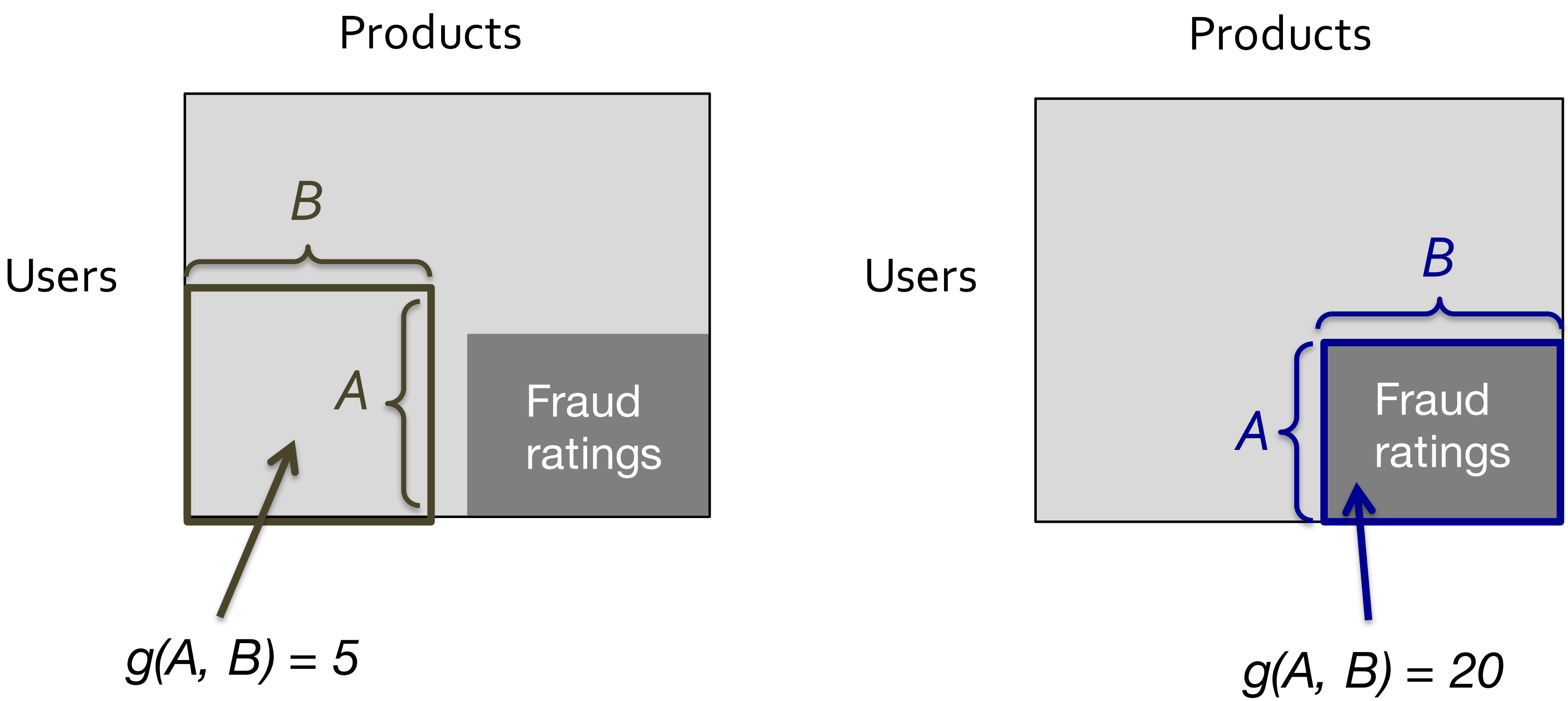
- Given:
  - Bipartite graph between users and products
  - May have prior node suspiciousness scores
- Develop detection metric that is:
  - **Camouflage-resistant**
  - **Near-linear time**
  - **Offers provable bounds**
  - **Works well in practice**



Hooi, et al. (KDD 2016)

Suspiciousness Metric

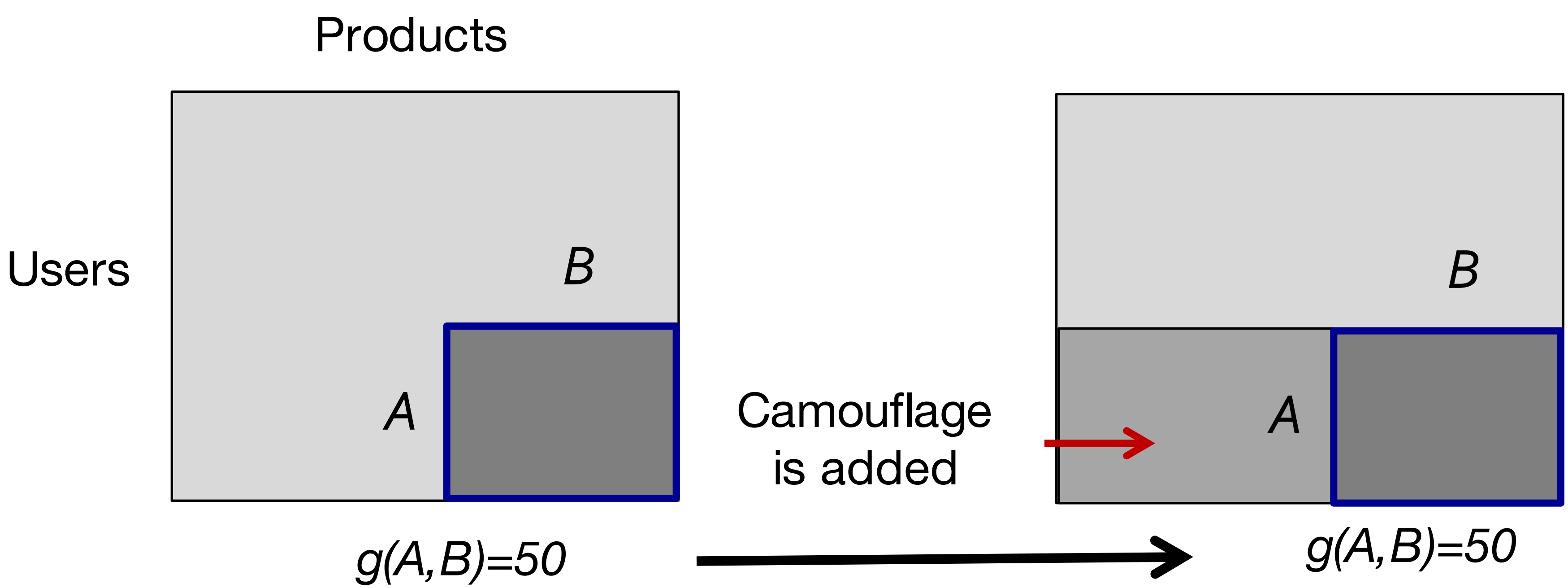
g(A,B) is a density metric for edges from set of users A to set of products B.



Hooi, et al. (KDD 2016)

Camouflage-Resistance

Metric g is ***camouflage-resistant*** if *g(A,B)* does not decrease when camouflage is added to *A*.

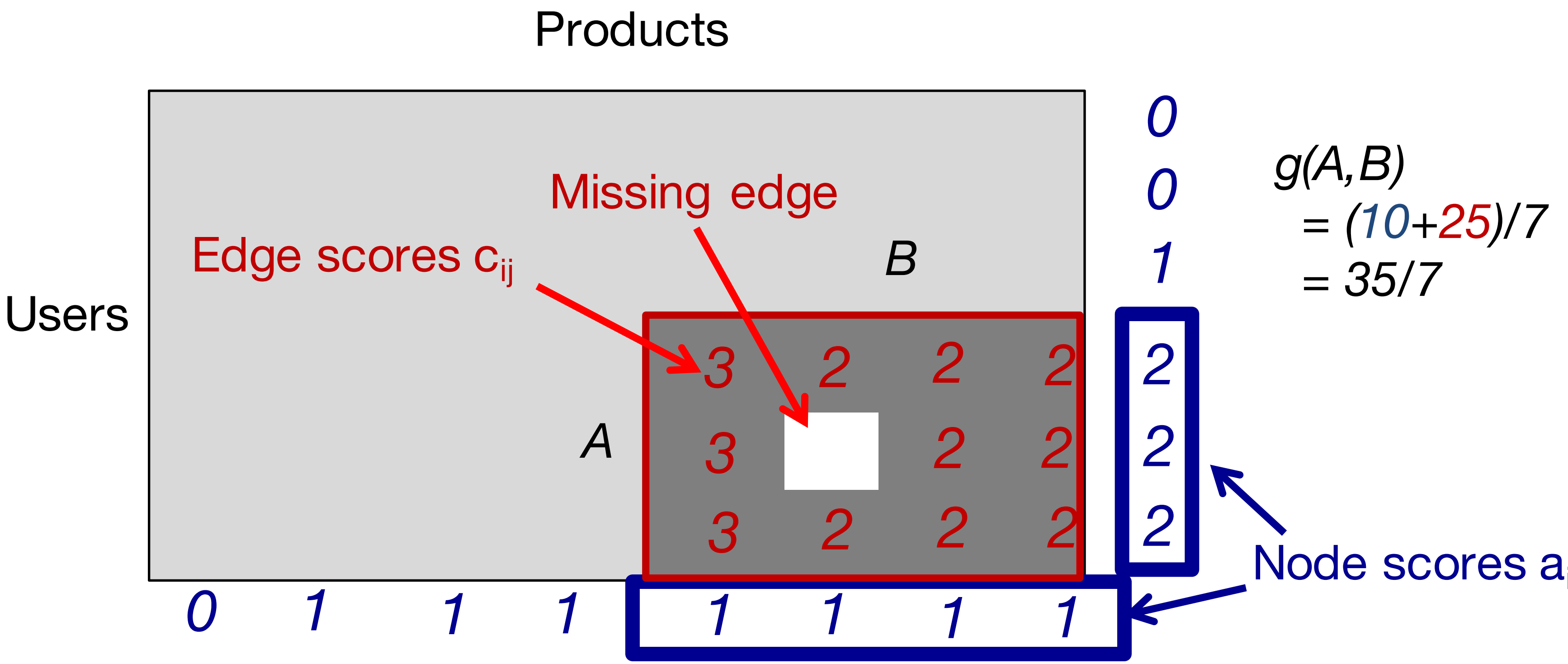


Hooi, et al. (KDD 2016)

Proposed Suspiciousness Metric

"Average suspiciousness" g(A,B)

$$= \frac{\text{(sum of node susp.)} + \text{(sum of edge susp.)}}{|A| + |B|}$$



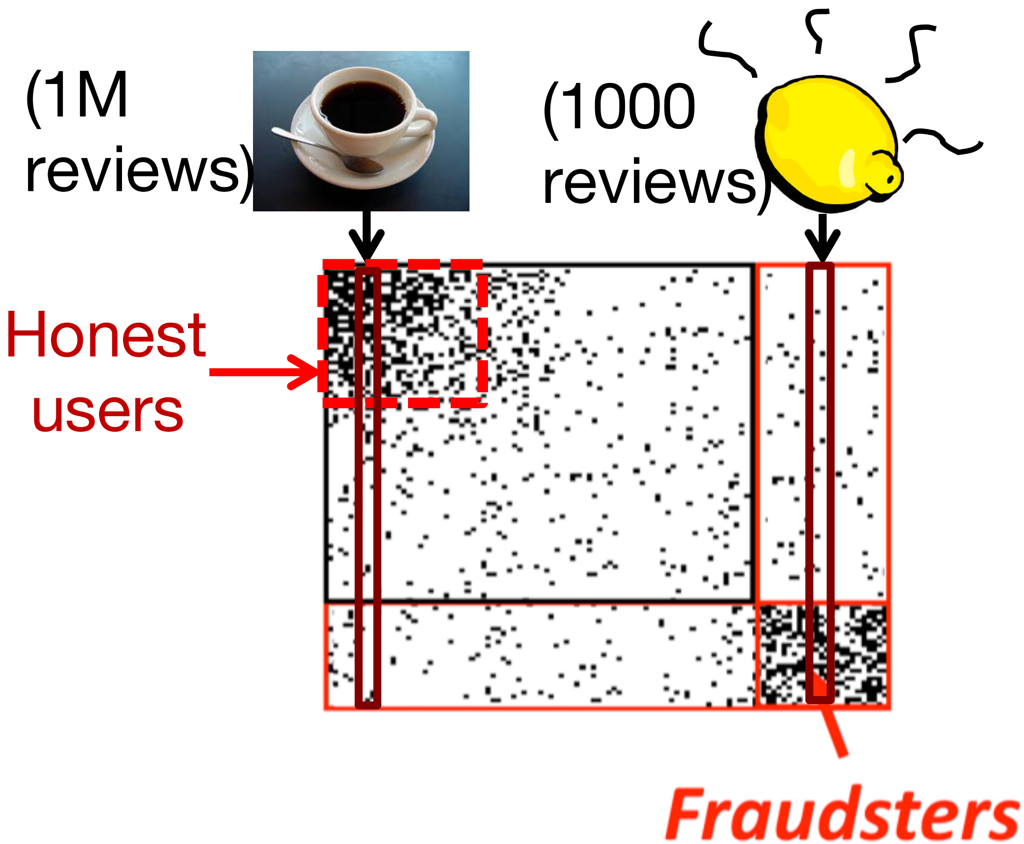
$g(A,B)$
$= (10+25)/7$
$= 35/7$

Edge Scores c_{ij}

tf-idf

Proposed weighting scheme:

c_{ij} = 1 / log(unweighted sum of j-th column)



Why?

- Popular products are not necessarily suspicious
- Fraudulent products have a high fraction of edges from fraudsters

Metric Properties

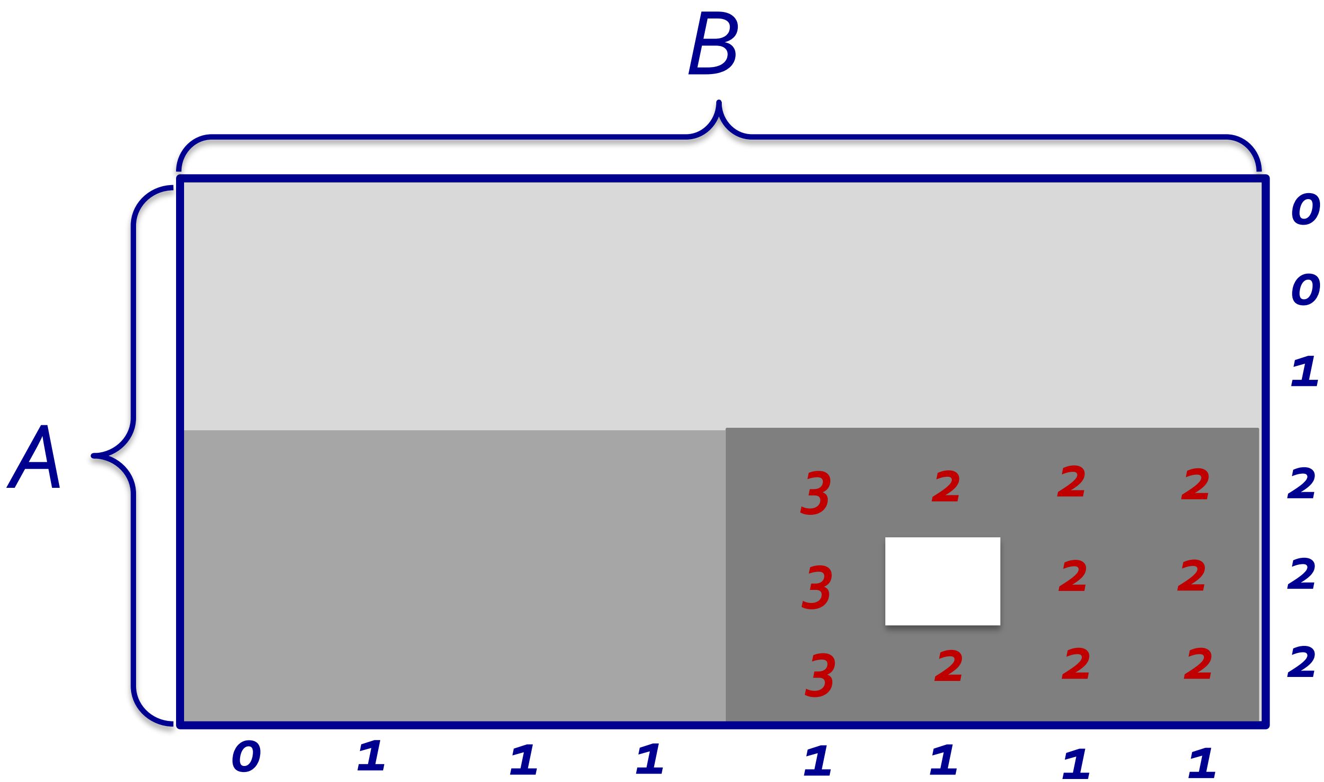
Average suspiciousness g(A,B):

- ☐ **Can be optimized in near-linear time**
- ☐ Provable bounds
- ☐ Camouflage-resistant
- ☐ Works in practice

Hooi, et al. (KDD 2016)

FRAUDAR: Greedy Algorithm

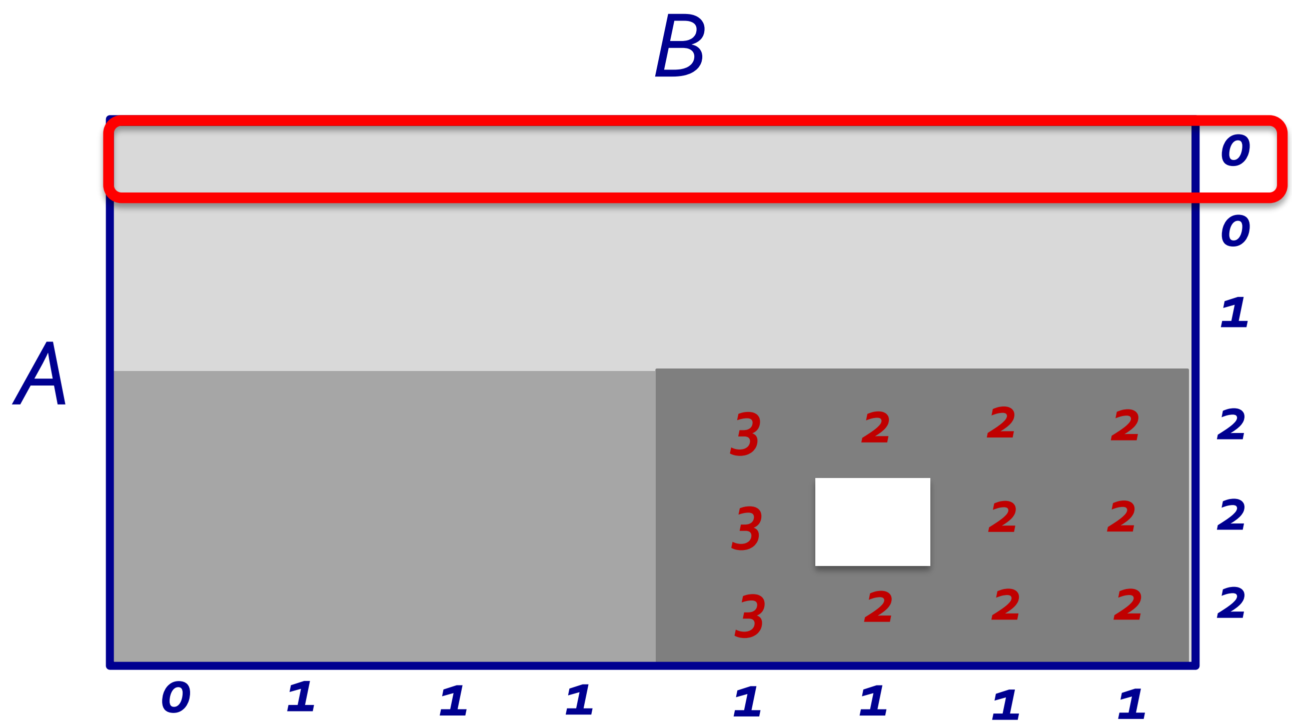
- Start with sets A, B as all users / products



Hooi, et al. (KDD 2016)

FRAUDAR: Greedy Algorithm (cont.)

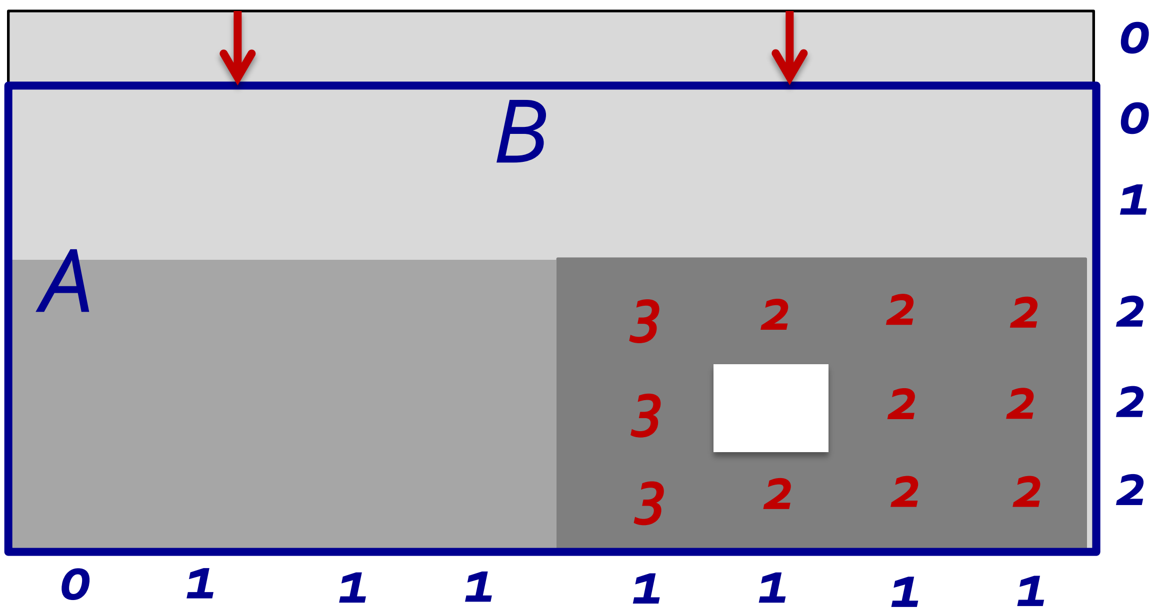
- Delete rows / columns greedily to maximize g (average suspiciousness)



Hooi, et al. (KDD 2016)

FRAUDAR: Greedy Algorithm (cont.)

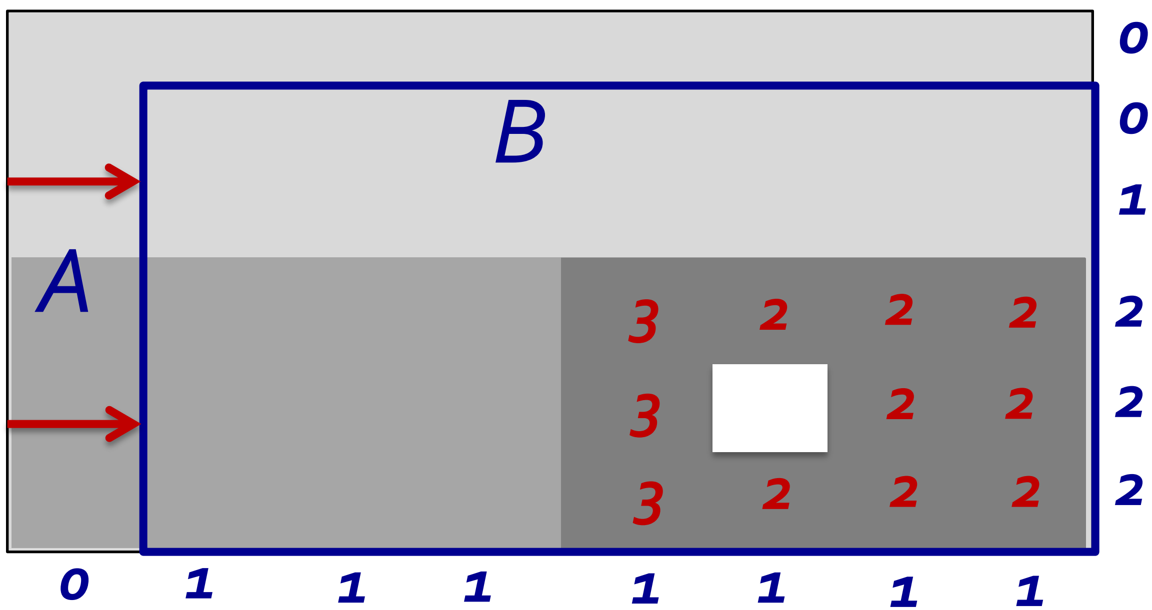
- Delete rows / columns greedily to maximize g (average suspiciousness)



Hooi, et al. (KDD 2016)

FRAUDAR: Greedy Algorithm (cont.)

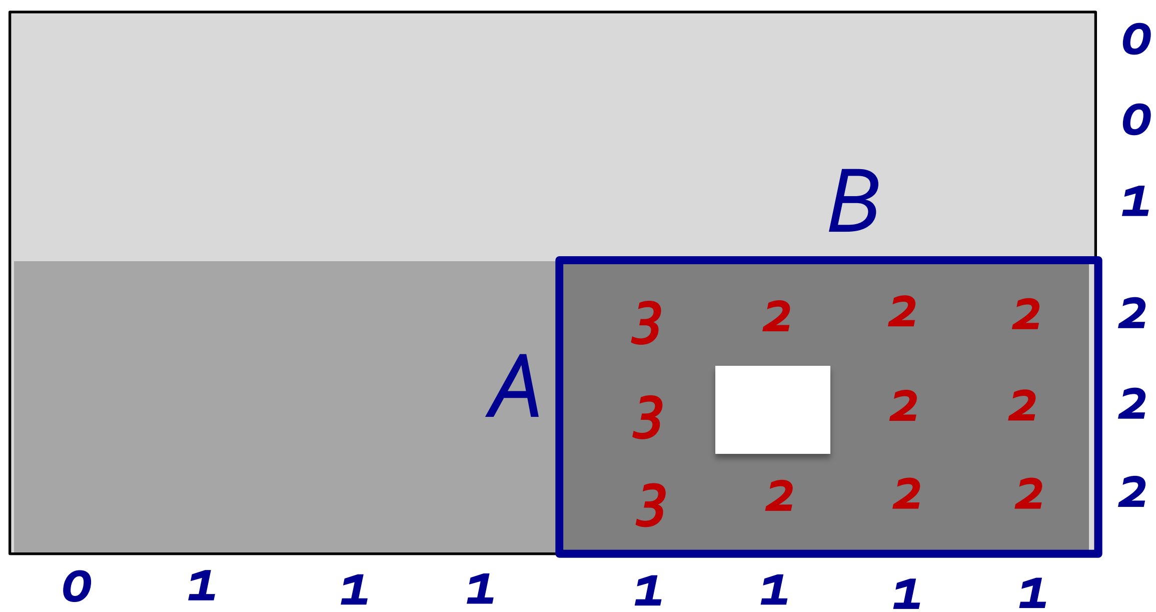
- Delete rows / columns greedily to maximize g (average suspiciousness)



Hooi, et al. (KDD 2016)

FRAUDAR: Greedy Algorithm (cont.)

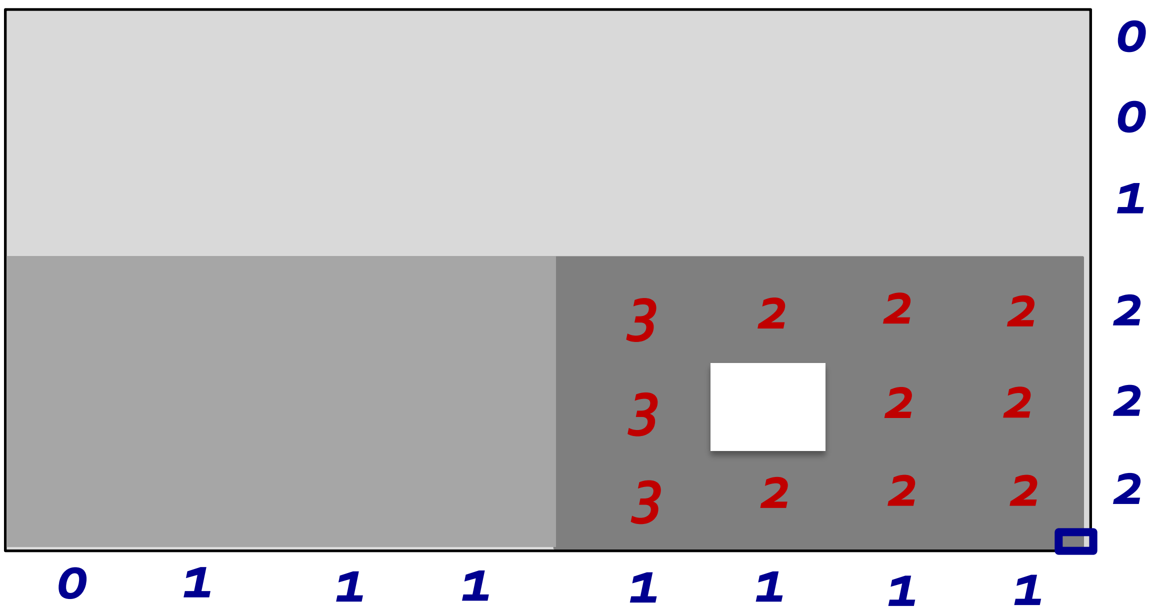
- Delete rows / columns greedily to maximize g (average suspiciousness)



Hooi, et al. (KDD 2016)

FRAUDAR: Greedy Algorithm (cont.)

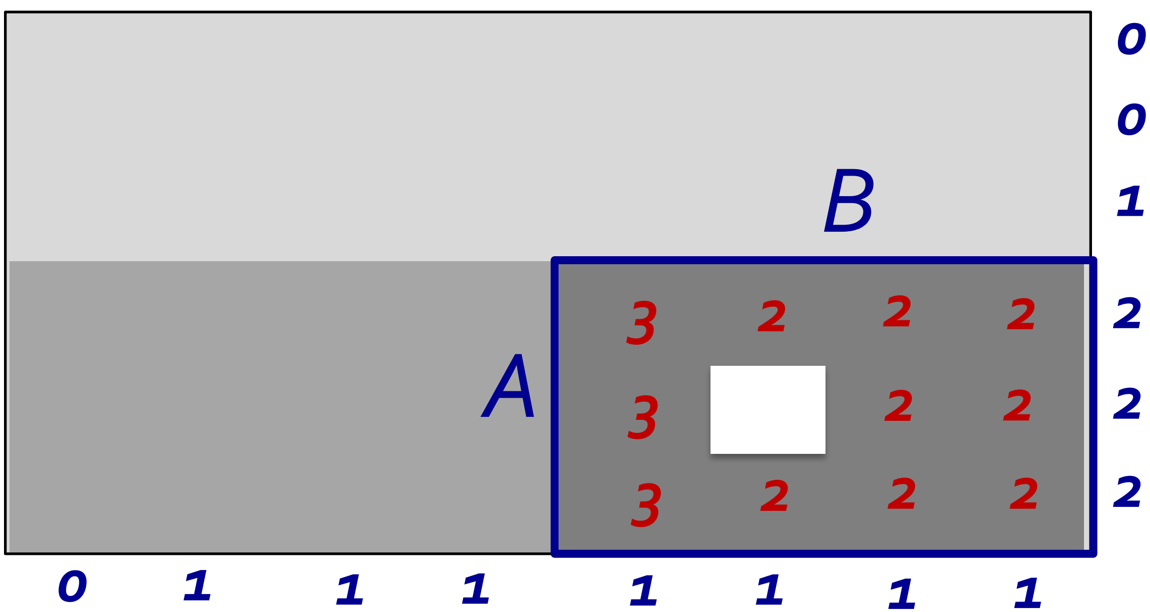
- Continue until A and B are empty



Hooi, et al. (KDD 2016)

FRAUDAR: Greedy Algorithm (cont.)

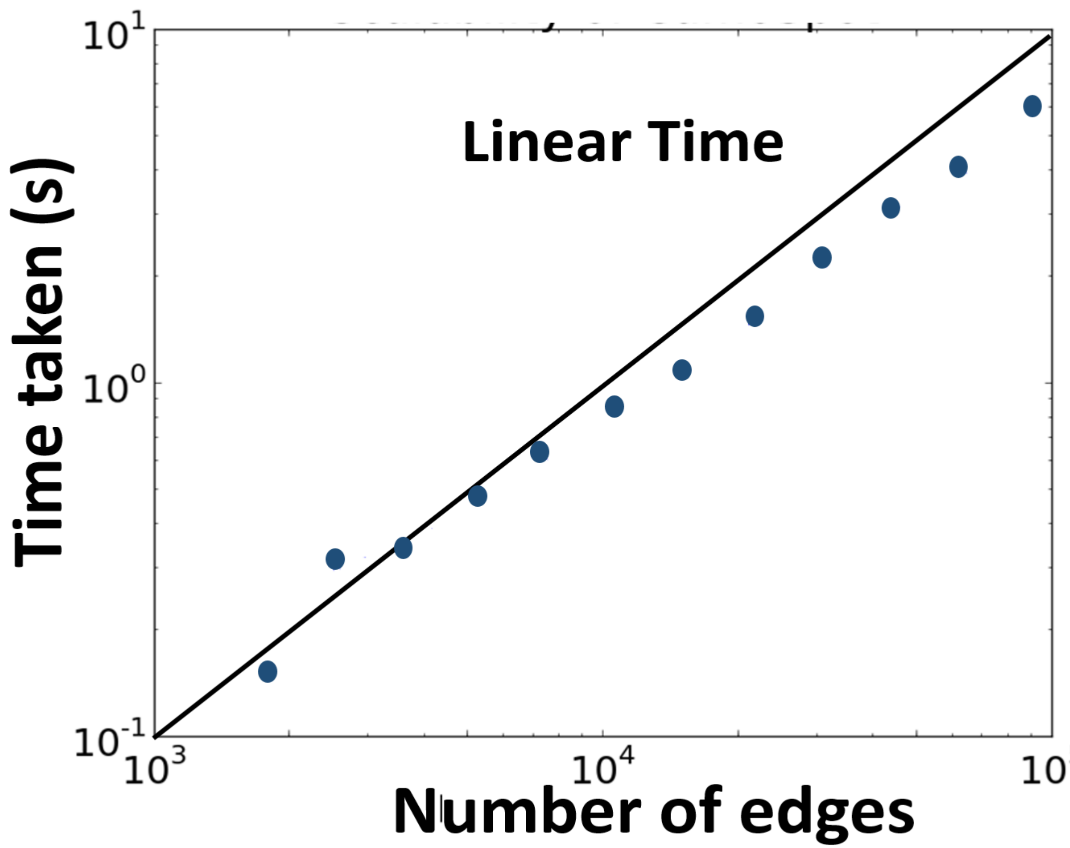
- Return the best subsets A and B seen so far (based on g)



Hooi, et al. (KDD 2016)

Computation Time

- O(|E| log(|V|)): using appropriate data structures



Hooi, et al. (KDD 2016)

Metric Properties

Average suspiciousness g(A,B):

- ☑ Can be optimized in near-linear time
- ☐ **Provable bounds**
- ☐ Camouflage-resistant
- ☐ Works in practice

Hooi, et al. (KDD 2016)

Theoretical Guarantee

- Theorem 1: The subgraph (A,B) returned by FRAUDAR satisfies

$$\boxed{g(\mathcal{A} \cup \mathcal{B}) \geq \frac{1}{2} g_{OPT}}$$

FRAUDAR subgraph

Optimum value of g

Hooi, et al. (KDD 2016)

Metric Properties

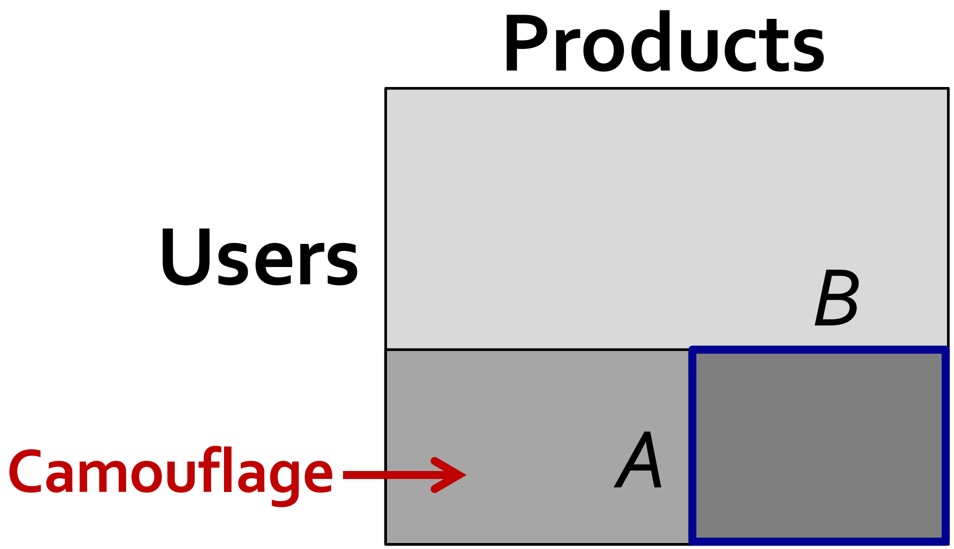
Average suspiciousness g(A,B):

- ☑ Can be optimized in near-linear time
- ☑ Provable bounds
- ☐ **Camouflage-resistant**
- ☐ Works in practice

Hooi, et al. (KDD 2016)

Camouflage Resistance

- Theorem 2: If c_{ij} is a *column weighting* (i.e. c_{ij} is any function of the j-th column), then **g is camouflage-resistant.**



(c_{ij} = 1 / log(sum of j-th column) satisfies this)

Hooi, et al. (KDD 2016)

Metric Properties

Average suspiciousness g(A,B):

- ☑ Can be optimized in near-linear time
- ☑ Provable bounds
- ☑ Camouflage-resistant
- ☐ **Works in practice**

Hooi, et al. (KDD 2016)

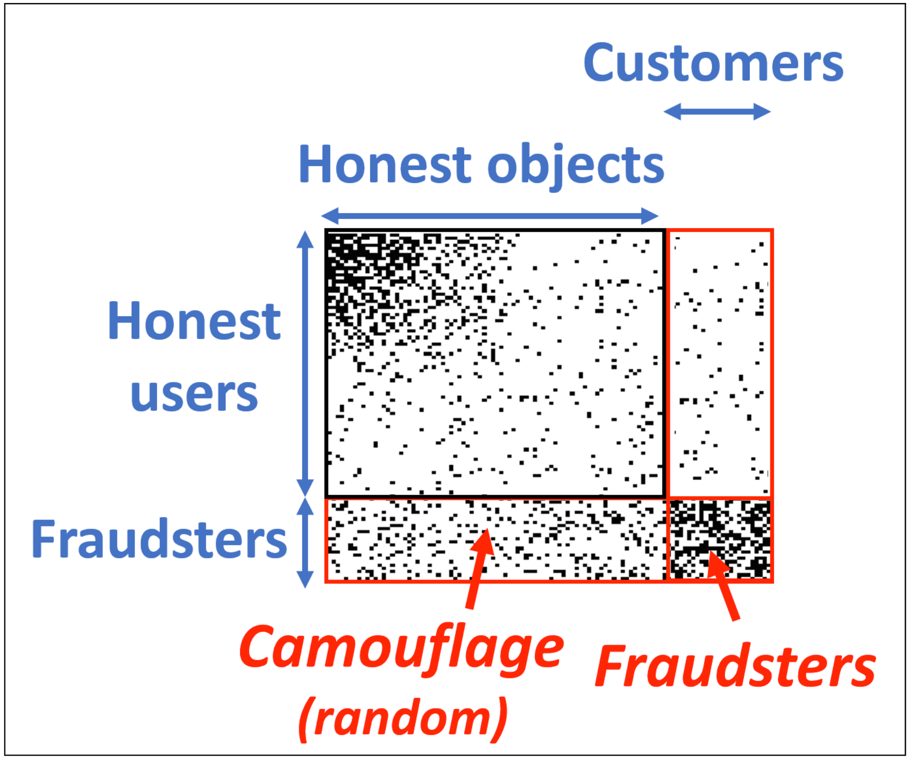
Experiments: Detecting Injection of Various Types of "Camouflage"

- Amazon Review Graph: 24K users, 4K products
- Injected 200 x 200 blocks with various types of camouflage
  - None
  - Random camouflage
  - Biased camouflage
  - Hijacked accounts

Hooi, et al. (KDD 2016)

Experiments: Detecting Injection of Various Types of "Camouflage"

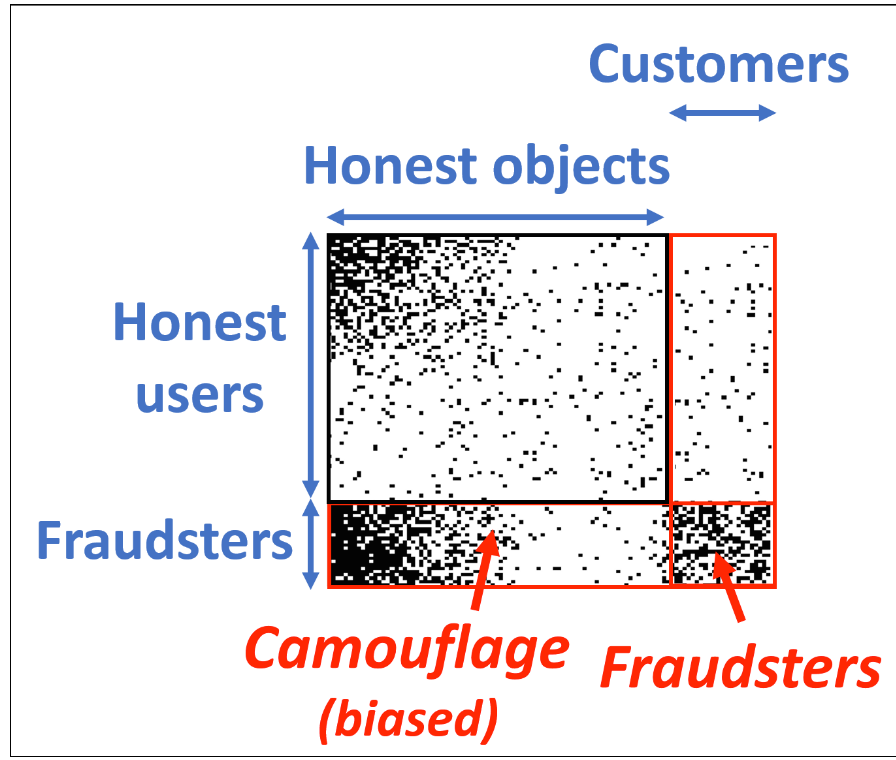
- Amazon Review Graph: 24K users, 4K products
- Injected 200 x 200 blocks with various types of camouflage
  – None
  – **Random camouflage**
  – Biased camouflage
  – Hijacked accounts



Hooi, et al. (KDD 2016)

Experiments: Detecting Injection of Various Types of "Camouflage"

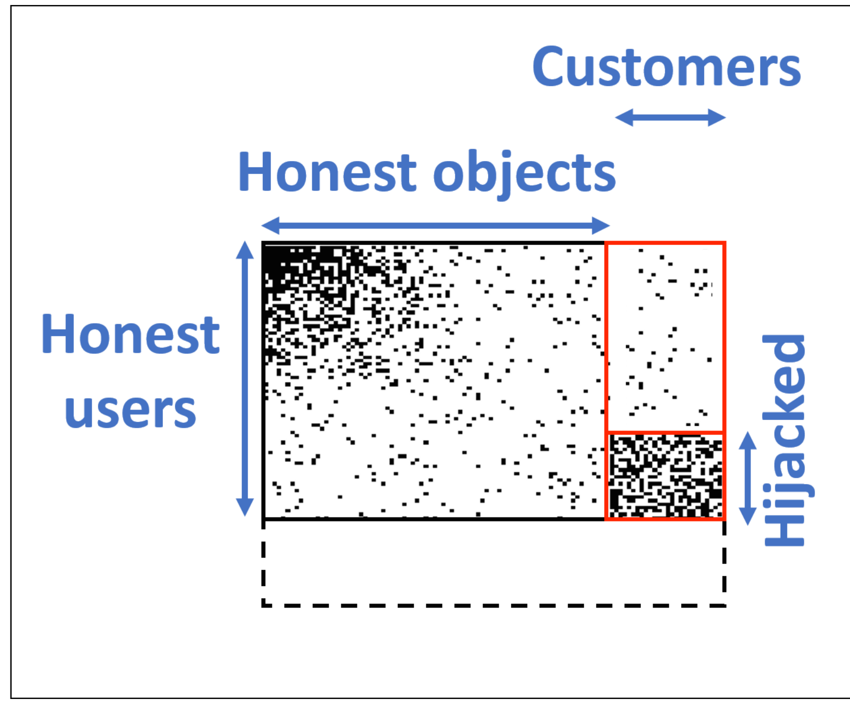
- Amazon Review Graph: 24K users, 4K products
- Injected 200 x 200 blocks with various types of camouflage
  - None
  - Random camouflage
  - **Biased camouflage**
  - Hijacked accounts



Hooi, et al. (KDD 2016)

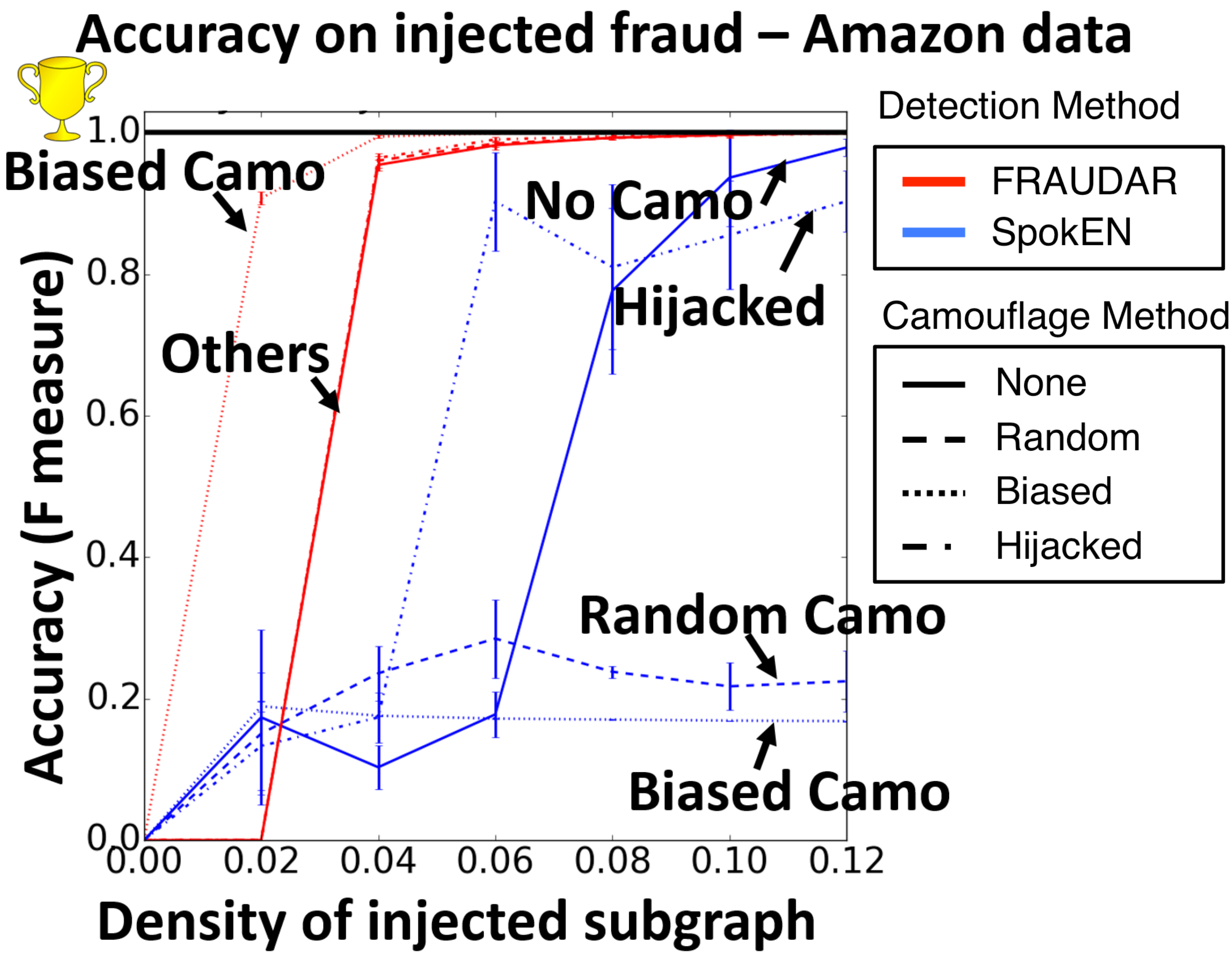
Experiments: Detecting Injection of Various Types of "Camouflage"

- Amazon Review Graph: 24K users, 4K products
- Injected 200 x 200 blocks with various types of camouflage
  - None
  - Random camouflage
  - Biased camouflage
  - **Hijacked accounts**



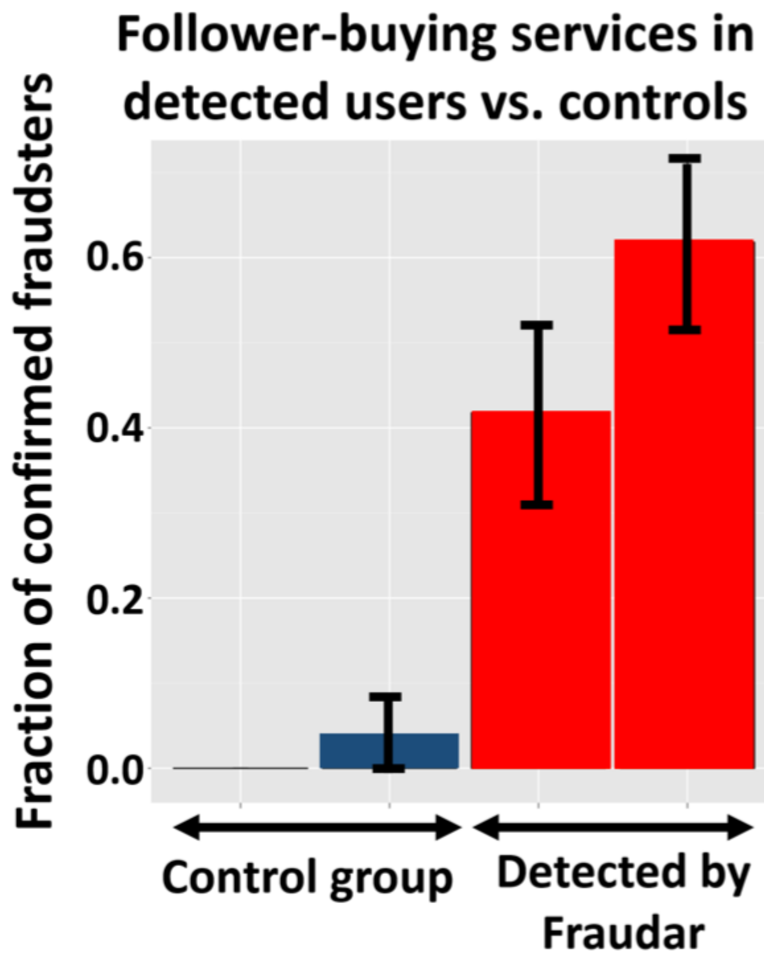
Hooi, et al. (KDD 2016)

Accuracy on Detecting Injected Fraud



Accuracy on Detecting Fraud in Real Twitter Data

- Found 4031x4313 size block of followers-followees with 68% density
- Users detected as fraudulent by Fraudar are more likely to be deleted, suspended, use Twitter user buying services.



Summary

- Spectral methods
  - Spectral clustering and community detection
- Spectral subspaces and spectral subspace plots
- EigenSpokes (singular vectors and "spokes")
- LockInfer ("**camouflage**", "fame", "pearls", "staircase", etc.)
- fBox (small-scale, stealthy attacks; reconstructed degrees)
- FRAUDAR (theoretical guarantees for bounding graph fraud in the face of **camouflage**)
- Applications: Mobile calls, Twitter social network, "user-product" reviews

References

- Andrew Y. Ng, Michael I. Jordan, Yair Weiss. “On Spectral Clustering: Analysis and an algorithm”, NIPS 2001.
- Scott White and Padhraic Smyth. “A spectral clustering approach to finding communities in graphs”, SDM 2005.
- M. E. J. Newman. “Finding community structure in networks using the eigenvectors of matrices”, Physical Review E 2006.
- Aditya Prakash, Mukund Seshadri, Ashwin Sridharan, Sridhar Machiraju, Christos Faloutsos. “EigenSpokes: Surprising Patterns and Scalable Community Chipping in Large Graphs”, PAKDD 2010.
- Meng Jiang, Peng Cui, Alex Beutel, Christos Faloutsos, Shiqiang Yang. “Inferring lockstep behavior from connectivity pattern in large graphs”, KAIS 2016.
- Neil Shah, Alex Beutel, Brian Gallagher, Christos Faloutsos. “Spotting suspicious link behavior with fBox: An adversarial perspective”, ICDM 2014.
- Bryan Hooi, Hyun Ah Song, Alex Beutel, Neil Shah, Kijung Shin, Christos Faloutsos. “FRAUDAR: Bounding Graph Fraud in the Face of Camouflage”, KDD 2016 Best Research Paper Award.

KDD 2017
Halifax, Nova Scotia - Canada
August 13 - 17, 2017

Tutorial: Data-Driven Approaches towards Malicious Behavior Modeling

Meng Jiang
University of Notre Dame

Srijan Kumar
Stanford University

Christos Faloutsos
Carnegie Mellon University

V.S. Subrahmanian
University of Maryland, College Park

Tutorial link: http://bit.ly/kdd2017